2018

Automatic Segmentation of Brain Tissues in Functional MRI

Sui Paul Ang
University of Wollongong

Automatic Segmentation of Brain Tissues in Functional MRI

A thesis submitted in partial fulfilment of the requirements
for the award of the degree

Master of Engineering by Research

from

University of Wollongong

by

Sui Paul Ang

School of Electrical, Computer and Telecommunications
Engineering

October 2018

Statement of Originality

I, Sui Paul Ang, declare that this thesis, submitted in partial fulfilment of the requirements for the award of Master of Engineering by Research, in the School of Electrical, Computer and Telecommunications Engineering, University of Wollongong, is wholly my own work unless otherwise referenced or acknowledged. The document has not been submitted for qualifications at any other academic institution.

Sui Paul Ang

1 October, 2018

Contents

List of Figures

List of Tables

Acronyms

MRI	Magnetic resonance imaging
fMRI	Functional magnetic resonance imaging
TR	Repetition time
TE	Echo time
$\mathbf{T}_{1w}$	T1 weighted
EPI	Echo planar imaging
GM	Gray matter
WM	White matter
BV	Blood vessel
NB	Non-brain
CSF	Cerebrospinal fluid
1D	One dimensional
2D	Two dimensional
3D	Three dimensional
4D	Four dimensional

k-NN	K-nearest neighbors
LSTM	Long short-term memory
CNN	Convolutional neural network
LRCN	Long-term recurrent convolutional network
STC	Slice scan time correction
DTW	Dynamic time warping
PCA	Principal component analysis
DSC	Dice similarity coefficient
pdf	Probability density function

Abstract

Accurate segmentation of different brain tissue types is the first step of understanding the neuronal activity in functional magnetic resonance imaging (fMRI). Due to the low spatial resolution of fMRI data and the absence of an automated segmentation approach, human experts require high resolution structural MRI images, which the fMRI data are superimposed on for analysis. The recent advent of high-resolution fMRI, along with temporal characteristic of fMRI data, suggests the possibility of segmenting fMRI image without relying on the high resolution structural MRI image.

This thesis proposes a patch-wise deep learning segmentation method using long-term recurrent convolutional network architecture. The proposed method comprises of three stages: spatial feature extraction with convolutional neural network, temporal feature extraction with long short-term memory, and brain tissue class prediction with *softmax* classifier.

The proposed method aims to segment five classes in fMRI images, which are gray matter, white matter, blood vessel, non-brain and cerebrospinal fluid. It achieves an average Dice similarity coefficient of 76.99%, which demonstrates that the proposed deep network could be used by specialists for segmenting fMRI data.

Acknowledgments

I would like to thank my supervisors, Assoc. Prof. Son Lam Phung, Dr. Mark Schira, and Prof. Abdesselam Bouzerdoum, for giving me the opportunity of pursuing my Master's degree under their guidances. I am very thankful for their support, feedback and encouragement throughout the study.

I am also grateful to my family members, colleagues and friends, who have given me support and advice during my study.

Chapter 1

Introduction

Chapter contents

1.1 Research objectives

Functional magnetic resonance imaging (fMRI), an extension of magnetic resonance imaging (MRI), has become an important tool in clinical diagnosis and brain research. Specialists use fMRI to non-invasively record changes in cortical activity. The fast acquisition time is key for fMRI, where echo planar imaging (EPI) sequence is used to scan brain images rapidly (1-4 seconds). To evoke controlled brain activity, scanning is performed while the subject is exposed to a predefined visual or auditory stimulus. The EPI (or fMRI) images collected across time are then analyzed to study the neural activity in response to the stimulus [10].

The EPI scans are typically low in spatial resolution (2.5-4 mm cube per voxel). To reveal anatomical structures, MRI typically uses a T1 weighted (T_{1w}) sequence that can capture high resolution brain images (1 mm cube per voxel or better), but with slow acquisition time (4-10 minutes) as a trade-off. A popular method to

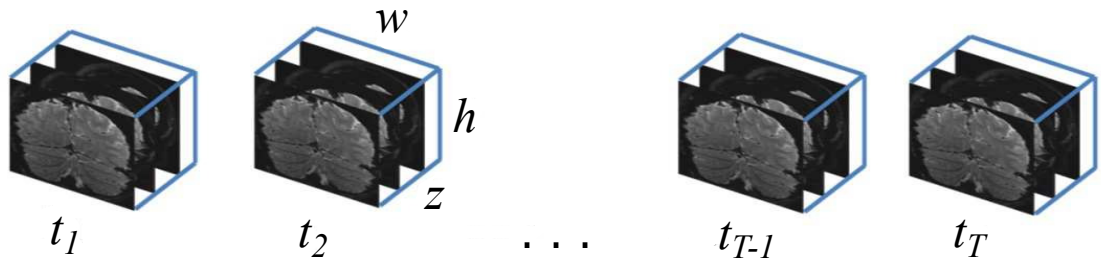


Figure 1.1: An example of fMRI data in an experimental run. In this figure, there are n brain volumes. Each brain volume is made up of $h \times w \times z$ voxels.

determine the brain area where changes in brain activity occur is by overlaying EPI scans over a T_{1w} image [11]. However, a perfect alignment between the EPI and T_{1w} images is difficult to achieve because of geometrical distortions [12].

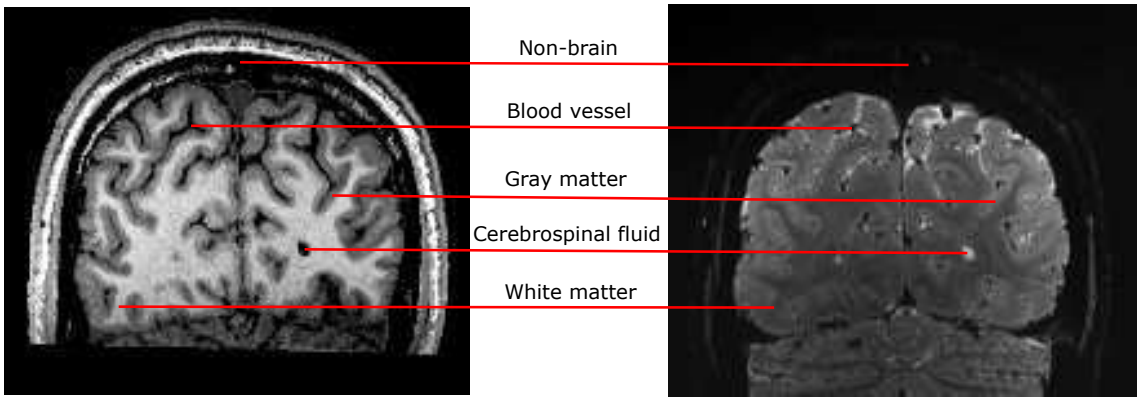
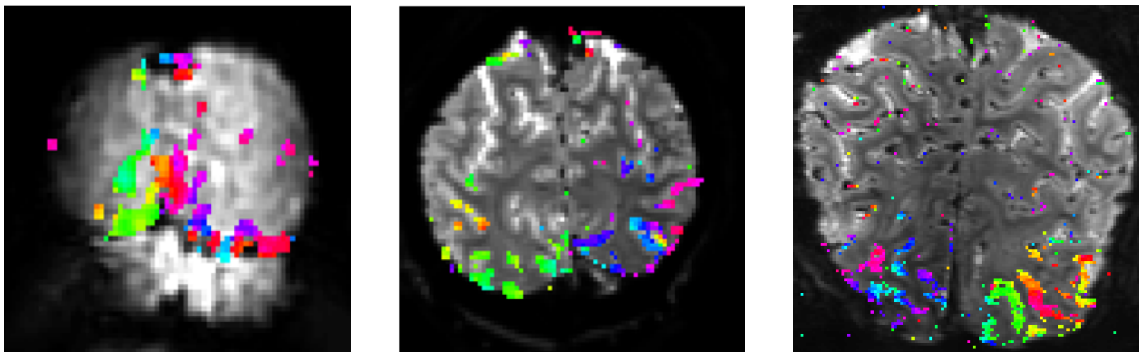


Figure 1.2: Differences between T_{1w} (left) and corresponding EPI (right) image. It is only possible to segment blood vessels in EPI image. In T_{1w} image, blood vessel and cerebrospinal fluid are not distinguishable.

With the recent advent of high-resolution fMRI (1.5 to 0.6 mm cube per voxel), the need for an accurate alignment is even higher. Traditionally, EPI data have a voxel (3D pixel) resolution of around 2.5 to 4 mm cube per voxel. A voxel of this size will still be roughly aligned to the correct tissue type in T_{1w} image, even with a misalignment as great as 1 mm cube per voxel. In contrast, for higher resolution data, e.g. 0.8 mm cube per voxel, as used in this thesis, such misalignment would be problematic.

However, high-resolution fMRI also provides better structural information. Moreover, fMRI data also contain temporal information that could be explored.



(a) $3 \times 3 \times 3$ mm^3 (b) $1.5 \times 1.5 \times 1.5$ mm^3 (c) $0.8 \times 0.8 \times 0.8$ mm^3

Figure 1.3: Differences in fMRI resolution.

Typically, the first step of understanding the neuronal activity is to classify (segment) each fMRI voxel into different brain tissue types, e.g. gray matter, white matter, blood vessel, non-brain and cerebrospinal fluid. While automatic segmentation (e.g. through Freesurfer [13]) has become more and more accepted as a standard for T_{1w} images, there is currently no automated approach for segmenting fMRI images. Thus, segmentation is often performed manually, which is tedious, error-prone, and subjective.

Deep learning methods are gaining popularity due to their success in various fields. For instance, convolutional neural network (CNN) is the current state-of-the-art in image processing field. It is often the main technique for image recognition and detection [14]. Another example is long short-term memory (LSTM). LSTM is a recurrent neural network (RNN) that can address the limitation of the traditional RNN in learning long-term dependency. LSTM is widely used for many sequence-related problems, such as speech recognition [15], human activity recognition [16], and time series classification [9].

Motivated by the above, the objective of this research is to develop a method based on deep learning for automatic segmentation of brain tissues in fMRI. Five types of brain tissue are considered: gray matter, white matter, blood vessel, non-brain, and cerebrospinal fluid.

This research aims to address the following questions:

1. Can deep learning methods outperform classical machine learning algorithms in segmenting brain tissue types from fMRI data?

2. Is temporal information sufficient for segmenting fMRI images? How useful is the temporal information of fMRI data?

3. How can deep learning methods utilize the spatio-temporal characteristic of fMRI data? Which deep neural network architecture is appropriate for this type of data?

1.2 Research contributions

The main contributions of this research are:

- An investigation of existing techniques for MRI brain tissue segmentation and fMRI signal classification.

- A novel automatic brain tissue segmentation method using deep learning. The proposed method is able to utilize the spatio-temporal characteristic of fMRI data.

- The first baseline score for fMRI brain tissue segmentation study.

The publication arose from this Masters research thesis is listed as follows:

- S. P. Ang, S. L. Phung, M. M. Schira, A. Bouzerdoum, S. T. M. Duong, "Human Brain Tissue Segmentation in fMRI using Deep Long-Term Recurrent Convolutional Network", *International Conference on Digital Image Computing: Techniques and Applications*, 2018. (Accepted: 21/09/2018)
  Abstract: Accurate segmentation of different brain tissue types is an important step in the study of neuronal activities using functional magnetic resonance imaging (fMRI). Traditionally, due to the low spatial resolution of fMRI data and the absence of an automated segmentation approach, human experts often resort to superimposing fMRI data on high resolution

structural MRI images for analysis. The recent advent of fMRI with higher spatial resolutions offers a new possibility of differentiating brain tissues by their spatio-temporal characteristics, without relying on the structural MRI images. In this paper, we propose a patch-wise deep learning method for segmenting human brain tissues into five types, which are gray matter, white matter, blood vessel, non-brain and cerebrospinal fluid. The proposed method achieves a classification rate of 84.04% and a Dice similarity coefficient of 76.99%, which exceed those by several other methods.

1.3 Thesis structure

The thesis is structured as follows:

- **Chapter 1** discusses the thesis objectives, and highlights the research contributions and publication.
- **Chapter 2** provides a brief introduction to MRI and fMRI, and reviews the existing works on MRI brain tissue segmentation and fMRI signal classification.
- **Chapter 3** introduces the fMRI dataset and the preprocessing steps.
- **Chapter 4** proposes a novel deep learning patch-wise fMRI brain tissue segmentation method.
- **Chapter 5** describes the experimental methods, and presents the experimental results. The experiments include hyperparameter tuning for the classifiers, as well as the evaluation of the proposed method along with other temporal domain and spatial domain classifiers.
- **Chapter 6** summarizes the research findings, and gives the future research directions and concluding remarks.

Chapter 2

Literature Review

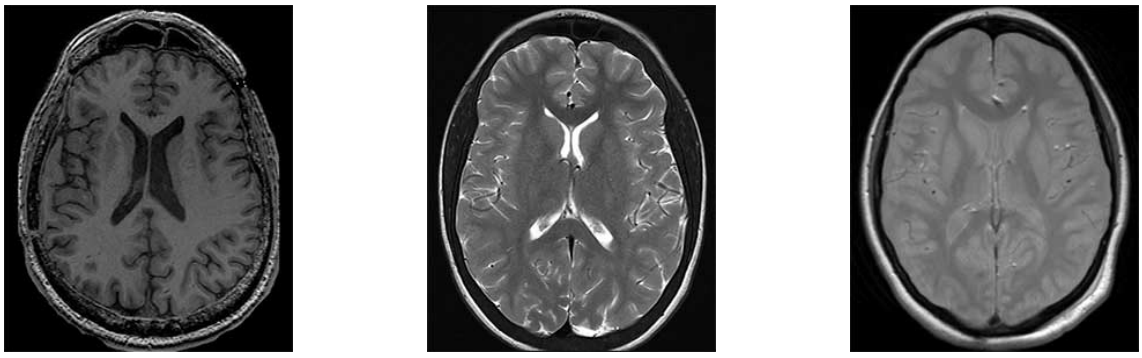
Chapter contents

This chapter presents a review of existing works on MRI brain tissue segmentation and fMRI signal classification. The chapter is organized as follows. Section 2.1 briefly introduces MRI and fMRI. Section 2.2 reviews existing works on MRI brain tissue segmentation while Section 2.3 reviews existing methods on fMRI signal classification.

2.1 A brief introduction to MRI and fMRI

Magnetic resonance imaging (MRI) [17] is an imaging technology that is capable of capturing and visualizing human anatomical structure through the use of a strong magnetic field. Initially, when a subject is in an MRI machine, the protons in the human body align with the magnetic field. Then, radio wave source is used to stimulate the protons, causing them to spin out of equilibrium, against the magnetic field. Once the radio wave source is cut off, the protons return to their initial state and emit radio wave signals. The time taken to return to its initial state is known as the relaxation time. The MRI scanner has receiver coils that will detect the radio wave signals, which form the MRI image.

To capture the 3D human brain, MRI typically scans 2D images slice-by-slice, which are then combined to form a complete 3D brain volume (see Figure 1.1). Each tissue type has a different relaxation time (such as T1, T2 and T2*), which produces different image contrast in MRI images. MRI uses a pulse sequence (e.g. T_{1w}, T_{2w}, and proton density weighted) that manipulates the relaxation time to enhance the appearance of certain tissues (see Figure 2.1).



(a) T_{1w} brain image. (b) T_{2w} brain image (c) Proton density weighted brain image

Figure 2.1: Example of T_{1w}, T_{2w}, and proton density weighted MRI images. Each MRI image has a different contrast. For example, the gray-white contrast of the T_{1w} image is inversed in the T_{2w} image. (Images are taken from [1])

Functional magnetic resonance imaging (fMRI) [18] is an extension of MRI that is capable of recording changes in brain activity. fMRI does not detect the

brain activity directly; instead it detects the effect of increased neuronal activity. In the human body, haemoglobin transports oxygen. There is a varying ratio of oxyhaemoglobin to deoxyhaemoglobin in the blood stream. Oxyhaemoglobin is isomagnetic whereas deoxyhaemoglobin is paramagnetic. The increase of neuronal activity will increase the blood flow because the brain requires more oxygen. However, this blood flow increases more than necessary, thus transporting more oxyhaemoglobin than required [19]. As a result, there will be a higher ratio of oxyhaemoglobin to deoxyhaemoglobin in the area of neuronal activation. This causes an increase of MRI signal intensity in the area of activation. This effect is known as the blood oxygenation level dependent (BOLD) response. In summary, fMRI is a method that records changes in neuronal activity indirectly by detecting the BOLD responses.

2.2 MRI brain tissue segmentation

According to the recent surveys by Akkus *et al.* [20] and Litjens *et al.* [21], deep learning techniques are widely used in literature. Application of deep learning in the medical domain has increased rapidly from around 10 papers in 2013 to around 225 papers in 2016 [21]. Generally, deep learning segmentation methods can be divided into two major approaches: (i) patch-wise [2, 3, 4, 22], and (ii) semantic [5, 23].

2.2.1 Patch-wise segmentation of MRI images

Patch-wise segmentation uses the standard CNN architecture. It can be seen as a special case of an image classification problem. Patch-wise segmentation labels each pixel in an MRI image by processing a rectangular region (i.e. a patch) centered on the pixel. Due to its simplicity, it is currently the most popular approach.

However, there are some disadvantages for this approach. First, there are

redundant computations because the extracted patches overlap. Second, spatial information that is available to the network is constrained by the patch size.

2.2.1.1 Patch-wise CNN with three modalities

Zhang *et al.* proposed a 2D patch-wise CNN to segment brain tissues (gray matter, white matter and cerebrospinal fluid) from MRI images of infants [2]. The proposed CNN accepts patches of size 13×13 pixels from three modalities: T1 weighted (T_{1w}), T2 weighted (T_{2w}) and fractional anisotropy. Their CNN architecture is unusual because it has three consecutive convolutional layers, and no pooling layer (see Figure 2.2). The proposed CNN has 5,332,995 trainable parameters.

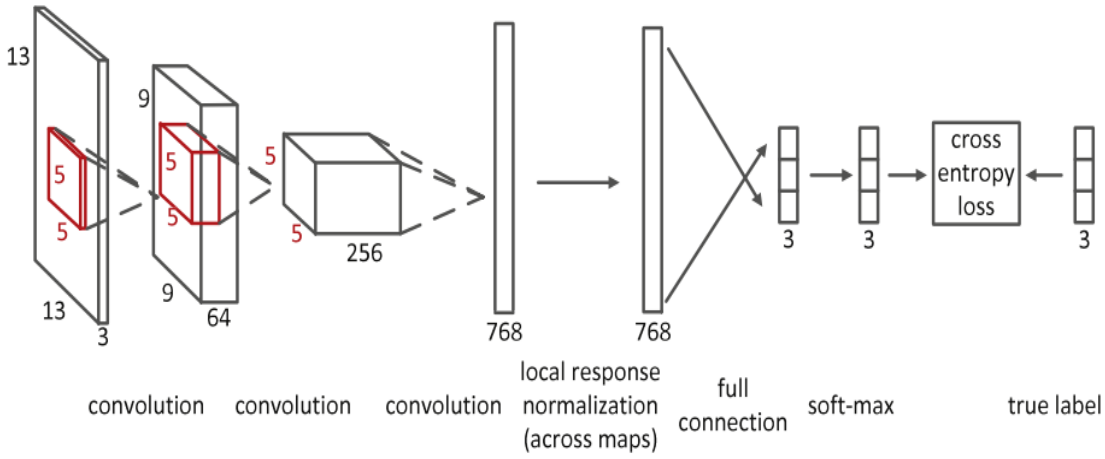


Figure 2.2: The deep CNN architecture used by Zhang *et al.* [2].

Before training, the T_{2w} and fractional anisotropy images were rigidly aligned to the corresponding T_{1w} images, and up-sampled to the same resolution as the T_{1w} images. Furthermore, the T_{1w} and T_{2w} images were intensity inhomogeneity corrected. The skull, brain stem and cerebellum were also removed from all three types of image. The network was trained using the stochastic gradient descent algorithm with the cross-entropy loss function. The network was trained for 370 epochs. To prevent overfitting, they applied dropout with 50% probability to the fully connected layer.

From the experiments, Zhang *et al.* discovered that T_{1w} image contains features that are useful for differentiating all three tissue types. The experiments also showed that fractional anisotropy image contains useful features for segmenting GM and WM, while T_{2w} image contains features that are useful for segmenting CSF. Combining all three modalities have a better segmentation performance than using any of the modalities individually. Zhang *et al.*'s experiments also indicated that the proposed CNN outperformed support vector machine and random forest classifier, suggesting that deep learning is able to extract more reliable features.

2.2.1.2 Multi-scale patch-wise CNN with 2D patches

Moeskops *et al.* [3] developed a multi-scale patch-wise CNN for MRI brain tissue segmentation. Because patch size is an important parameter in the patch-wise segmentation approach, the proposed multi-scale CNN accepts input of different sizes in parallel, thereby capturing different types of information. The larger patch provides more spatial information whereas the smaller patch provides more detailed information about the local surrounding voxels.

Moeskops *et al.* used patch sizes of 25×25, 51×51 and 75×75 pixels. Their architecture can be considered as three CNNs, which are trained separately and then joined at the last layer for class prediction. Each CNN is customized according to its input (e.g. a larger filter is used for a larger patch).

The MRI images were bias corrected and the intensities were scaled to the $[0, 1023]$ range. Brain area masks were also generated to remove the non-brain voxels. The network was trained using the RMSProp optimizer with the cross-entropy loss function for 10 epochs. The proposed method was evaluated on three datasets: neonatal images, ageing adult images, and young adult images.

The experiments performed by Moeskops *et al.* demonstrated that the smallest patch size, i.e. 25×25 pixels, is able to capture local texture accurately but fails on spatial consistency. In contrast, the largest patch size, i.e. 75×75 pixels, produces smooth segmentation, but misses small details.

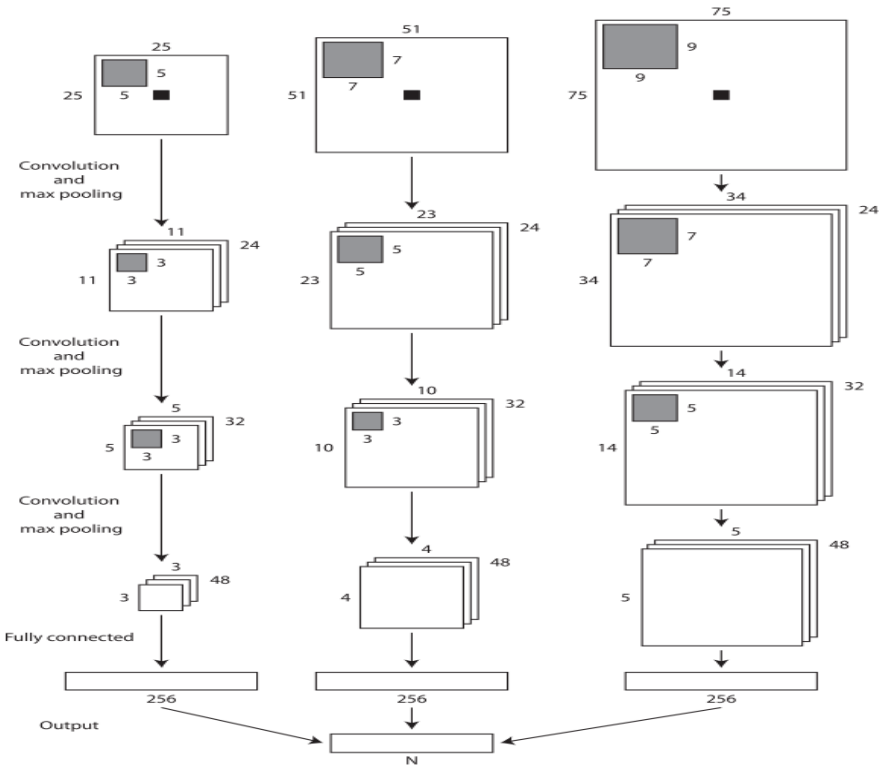


Figure 2.3: The multi-scale CNN architecture used by Moeskops *et al.* [3].

2.2.1.3 Multi-scale patch-wise CNN with 3D and 2D patches

Brebisson *et al.* proposed a multi-scale patch-wise CNN for 3D and 2D patches [4]. This architecture considers the 3D nature of MRI data. It has 8 input pathways in total. The first seven input pathways are image patches, which are fed into CNNs for feature extraction. The last input pathway is distance-to-centroid features.

The first three CNNs accept 2D patches of size 29×29 pixels from sagittal, coronal and transverse planes. Similarly, the next three CNNs accept three downscaled 2D patches from the same three planes. Each 2D patch of size 87×87 pixels is downscaled by a factor of 3 to a size of 29×29 pixels. The downscaling is performed using a mean pooling operation. By using a lower resolution, the network can capture larger spatial context while having lower memory requirement and lesser computational complexity. The seventh CNN accepts 3D patches of size $13 \times 13 \times 13$ pixels.

For the last input pathway, the distance-to-centroid features are combined

with features from other pathways in the fully connected layer for processing. In their dataset, each MRI image has 134 anatomical regions. For region l, the centroid $C_l = (x, y, z)$ is defined as the center of all the uniformly weighed voxels of that region. The average distance between two centroids is the same for all brain images. Let M be the number of centroids, that is, $M = 134$. The average distance is calculated as

$$D = \frac{M \times (M+1)}{2} \sum_{i=1}^{M} \sum_{j=1}^{M} d(C_i, C_j), \tag{2.1}$$

where d is the Euclidean distance function. Because computing the M centroids during the training phase uses the ground-truth, it is not possible during testing. Hence, the authors proposed the following iterative procedure:

1. The proposed network, without the distance-to-centroid branch, is trained to obtain a coarse segmentation.

2. The full proposed network, which uses the coarse segmentation to predict the centroids, is trained to refine the segmentation.

3. The refined segmentation is used to obtain better approximation of the centroids.

Step 2 and 3 are repeated until the network converges. The network has a total of 30,565,555 parameters.

Brebisson *et al.* performed the experiments using the MICCAI 2012 Challenge dataset. They trained the network using the stochastic gradient descent algorithm. The loss function used here was a negative log-likelihood function:

$$Loss = -\frac{1}{N} \sum_{n=1}^{N} \log(y_n \cdot \hat{y}_n), \tag{2.2}$$

where N is the total number of samples, and y_n and $\hat{y}_n$ are the predicted and true output probability vectors for the n-th sample, respectively. The $\hat{y}$ is a one-

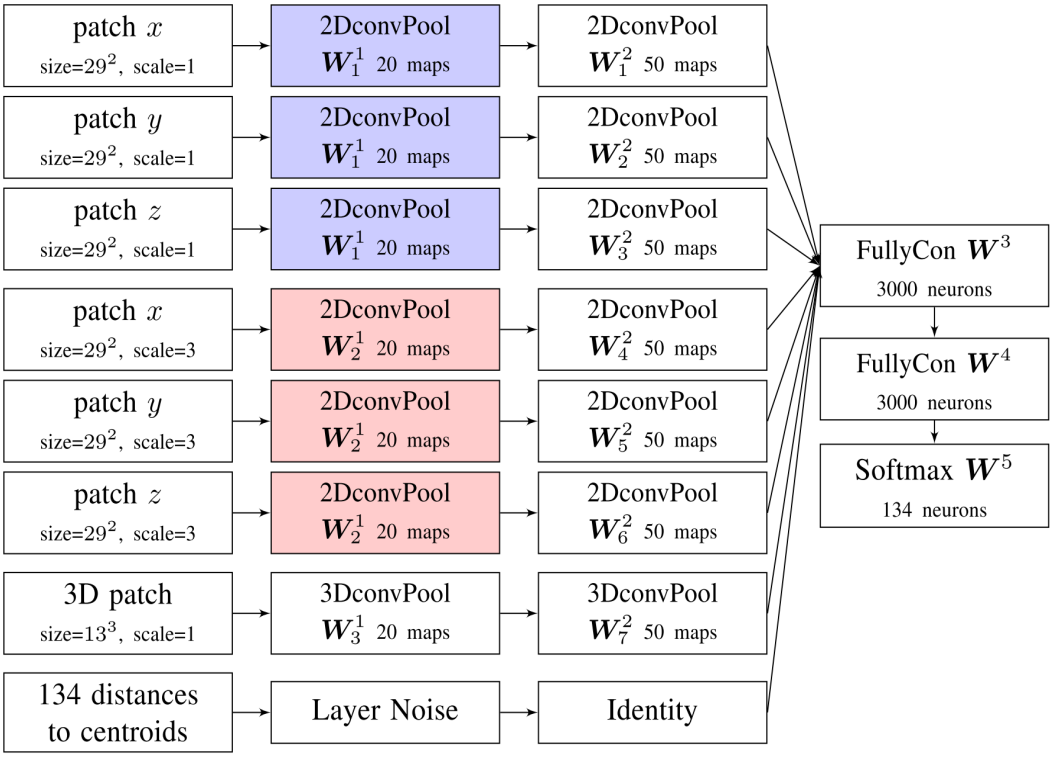


Figure 2.4: The multi-scale patch-wise CNN used by Brebisson *et al.* [4].

hot encoded vector (each index represents a class), where the true class index is denoted as one and the rest are zeros. During training, the distance-to-centroid features were artificially corrupted with Gaussian noise. The training was stopped early if the error rate of the validation set had not improved after 10 epochs.

The experiments performed by Brebisson *et al.* showed that distance-to-centroid features and downscaled 2D patches contain redundant information (using both feature types or using one feature type achieved a similar accuracy), thus choosing any one is sufficient. The experiments also showed that three orthogonal (sagittal, coronal and transverse planes) 2D patches are an excellent alternative to an individual 3D patch because they require less memory and can achieve competitive performance.

2.2.2 Semantic segmentation of MRI images

Semantic segmentation takes the entire MRI image as input, and generates directly a segmentation map (of the same size as the input) for all pixels in one forward pass. This allows the network to capture full contextual information of the image, rather than being constrained by the patch size. Unlike the patch-wise approach, there are no redundant overlapping pixels in this approach, resulting in a shorter computation time.

However, the lack of training samples is a common problem in a semantic approach. Unlike the patch-wise approach, a whole image is considered as one sample in this approach. Moreover, modelling 3D or 4D of fMRI data is challenging in a semantic approach due to memory and computational requirements.

2.2.2.1 Fully convolutional network

In [5], Nie *et al.* developed a fully convolutional network (FCN) model that segments MRI brain images semantically from three modalities (T_{1w}, T_{2w} and fractional anisotropy). Their architecture can be seen as three independent FCNs that are trained on each modality (see Figure 2.5). Then, the features from each FCN are joined in a later stage for class label prediction.

FCN architecture replaces the fully connected layers in CNN with convolutional layers or deconvolutional layers. The computation in the deconvolutional layer is the inverse of convolutional layer [24]. This computation up-samples the input instead of down-samples it. Up-sampling the input with a factor of w is the same as a convolution operation with a fractional input stride of $1/w$.

An FCN consists of two steps: down-sampling and up-sampling. The down-sampling step uses convolutional layers, whereas the up-sampling step uses deconvolutional layers. In Nie *et al.*'s proposed architecture (see Figure 2.5), the first, second and third groups are the down-sampling step, and the remaining groups are the up-sampling step. The total trainable parameters for this architecture are

96% smaller than Zhang *et al.*'s proposed patch-wise network, resulting of only 20,8548 parameters.

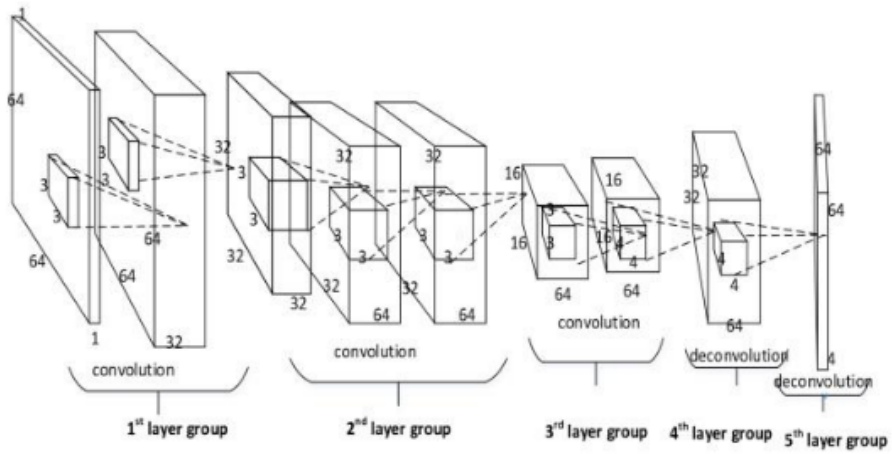


Figure 2.5: The FCN architecture used by Nie *et al.* for one modality [5].

The preprocessing steps performed on the data were the same as Zhang *et al.* [2] (described in Section 2.2.1.1). To tackle the class imbalance problem, Nie *et al.* applied weights to the cross-entropy loss function to penalize the minority classes more. The weights were initialized with the Xavier algorithm, and the biases were initialized to zero.

For the experimentation, Nie *et al.* compared with Zhang *et al*'s method using the same leave-one-out evaluation technique. Because the dataset was small, they performed the semantic segmentation on patches of size 64×64 pixels first, then the result of each patch was combined to form the final images.

The results show an improvement when compared to Zhang *et al.*'s method. The Dice similarity coefficients (DSCs) were 0.855 versus 0.835 for CSF, 0.873 versus 0.852 for GM, and 0.887 versus 0.864 for WM.

2.2.2.2 U-net

Another well known semantic segmentation architecture is the U-net. U-net was proposed by Ronneberger *et al.* [6], extending the concept of FCN, for a binary cell segmentation task (*positive* or *negative* class only). The U-net has an

equal amount of down-sampling and up-sampling steps, and skip connections between the down-sampling and up-sampling counterparts. A skip connection allows an up-sampling step to use directly the feature maps from its down-sampling counterpart; this strategy helps to retain useful spatial features [25]. These characteristics result in a U-shaped architecture (see Figure 2.6).

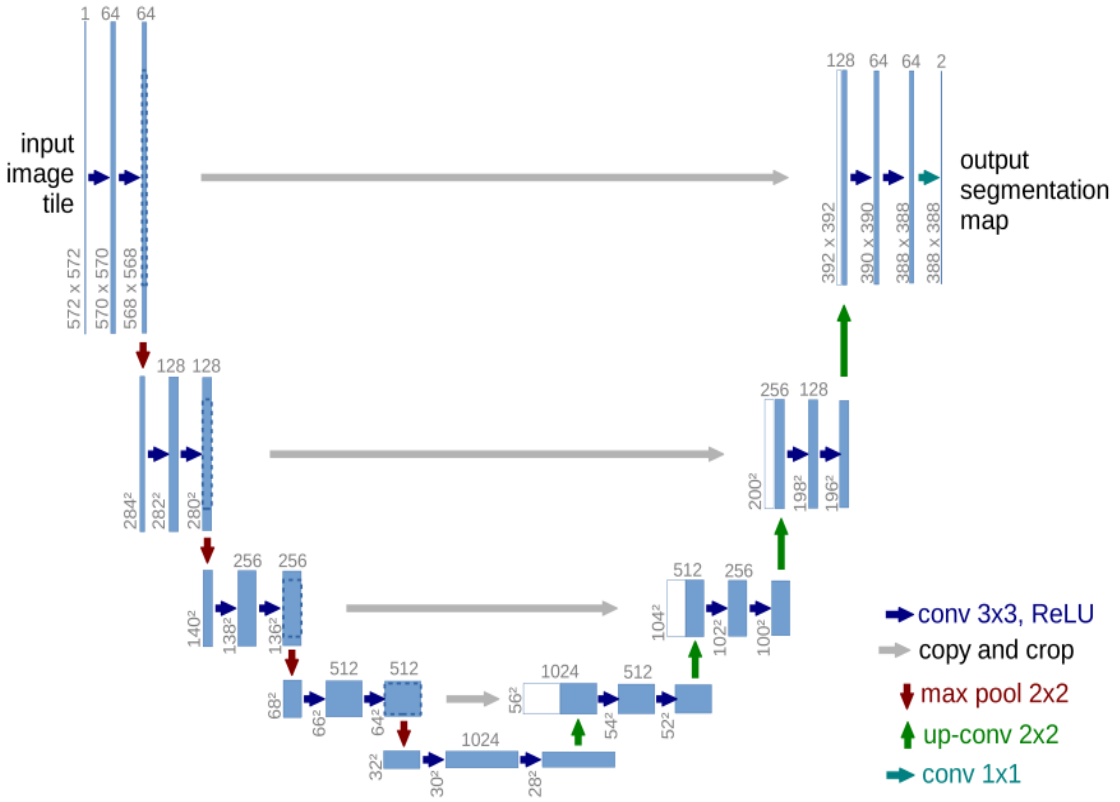


Figure 2.6: The U-net architecture [6].

Deng *et al.* further extended the U-net model to segment three classes (gray matter, white matter and cerebrospinal fluid) from 2D MRI images [23]. To allow a multi-class segmentation, Deng *et al.* replaced the cross-entropy loss function with the Dice similarity coefficient function:

$$L(y, \hat{y}) = 4 - \sum_{c=0}^{3} DSC(y_c, \hat{y}_c),$$
$$DSC = \frac{2TP}{2TP + FP + FN}, \tag{2.3}$$

where TP is the true positive, FP is the false positive, and FN is the false negative.

The network contains 31,030,788 parameters. They evaluated their method on the IBSR18 dataset, and obtained competitive results.

2.3 fMRI signal classification

Our literature analysis revealed that more recent fMRI works are focusing on fMRI signal classification, for example classifying the brain state of subjects (such as *performing task* or *idling* [26]), and detecting diseases (such as Attention Deficit Hyperactivity Disorder [7, 27, 28] and Alzheimers [29]).

Due to the high dimensionality of fMRI data, feature extraction is often needed. Traditionally, most fMRI signal classification methods first extract hand-crafted features and then apply a classifier such as support vector machine (SVM) and k-nearest neighbor classifier (k-NN). However, with the success of deep learning, recent approaches employ deep learning techniques where the feature extraction is automated.

2.3.1 Principal component analysis

Principal component analysis (PCA) is a non-parametric method for selecting relevant features from a dataset [30]. Because features with large variances contain important structures, PCA maps the original features into a new and smaller set of orthogonal features that maximize the variance.

Xie *et al.* used PCA to extract features for the SVM classifier [26], which predicts the brain states of subjects (*performing task* or *idle*). In their proposed method, an fMRI image X of size $N \times M$ is represented as a 2D matrix, where rows are features and columns are samples:

$$X = \begin{bmatrix} x_{11} & \dots & x_{1M} \\ \vdots & \ddots & \vdots \\ x_{N1} & \dots & x_{NM} \end{bmatrix}.$$

Before performing PCA, each feature needs to be normalized to have a mean of zero. To extract features using PCA, a covariance matrix of X is first calculated:

$$C_X = \frac{1}{M} X X^\top, \tag{2.4}$$

where C_X is size of $N \times N$. Then, C_X is decomposed into eigenvectors and eigenvalues. The eigenvalues are sorted with descending order, producing $\{\lambda_1, \lambda_2, \ldots, \lambda_N\}$. The corresponding eigenvectors $\{v_1, v_2, \ldots, v_N\}$ are known as principal components, with the v_1 being the most significant and v_N being the least significant.

To reduce N-dimensional features to K-dimensions, we select only the first K principal components, which forms $P = \{v_1, v_2, \ldots, v_K\}$. The fMRI image dimensionality is reduced to the K-dimension as follows:

$$Y = P^\top X, \tag{2.5}$$

where Y is the new image of size $K \times M$.

Xie *et al.*'s experiments showed that PCA extracted features reduced the training time of SVM significantly (from an average of 44 s to 0.75 s), while maintaining the classification performance. In fact, the classification accuracy of SVM was higher with PCA extracted features (96% versus 90%). This suggests that PCA is able to remove noise or redundant features from fMRI data.

2.3.2 Dynamic time warping

Meszlenyi *et al.* used dynamic time warping (DTW) distance as features, and SVM and LASSO are used as classifiers for gender and ADHD classification [27]. In their dataset, each subject has 90 functional regions of interest (ROI). Each ROI contains an averaged BOLD time series. For each subject, using the 90 time series, the authors compute a full connectivity matrix with DTW, which is fed into the classifiers.

DTW was pioneered in speech recognition [31], and since then it has been used in many other fields including medical engineering, finance, and image processing. DTW is an elastic distance measure that can minimize the effects of shifting and distortion in time [32]. DTW is described as follows.

Consider two fMRI signal time series $\mathbf{A} = \{a_1, a_2, ..., a_m\}$ and $\mathbf{B} = \{b_1, b_2, ..., b_n\}$ of length m and n, respectively. A distance matrix C of size $m \times n$ is first defined, where $C_{i,j} = (a_i - b_j)^2$. Then, an optimum warping path $\mathbf{W}$ is selected, which comprises of a set of coordinates in C that is chosen to define the mapping between time series $\mathbf{A}$ and $\mathbf{B}$. The optimum warping path is defined as the path that has the least cost. Additionally, let $\mathbf{W}_k = (i, j)$ and $\mathbf{W}_{k-1} = (i^{-1}, j^{-1})$, $\mathbf{W}$ must meet the following criteria:

1. *Boundary condition*: $\mathbf{W}$ must start at the bottom left corner $(1, 1)$ and end at the top right corner (m, n) of C.

2. *Continuity*: $i - i^{-1} \leq 1$ and $j - j^{-1} \leq 1$. Distance between adjacent points in $\mathbf{W}$ must only be lesser or equal to one.

3. *Monotonicity*: $i \geq (i^{-1})$ and $j \geq (j^{-1})$. $\mathbf{W}$ must traverse only either forward or to the adjacent cell.

Dynamic programming is used to find the optimum path by evaluating the recurrence of [33]

$$\gamma(i, j) = C_{i,j} + \min\left\{\gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1)\right\}, \tag{2.6}$$

where $\gamma(i, j)$ is the cumulative distances of the current cell $C_{i,j}$ and the minimum cumulative distances out of the three adjacent cells. DTW distance between time series $\mathbf{A}$ and $\mathbf{B}$ is defined as

$$DTW(\mathbf{A}, \mathbf{B}) = \sqrt{\sum_k \mathbf{W}_k}. \tag{2.7}$$

However, a limitation of DTW is that its computation cost of $O(n^2)$ is very high [34]. To improve the DTW computation speed, we can set an adjustment window λ to constrain the optimum warping path to the cells near the diagonal in C. This heuristic is established because the optimum path is always found near the diagonal [31].

Meszlenyi *et al.* demonstrated that DTW-based features had a higher classification performance than correlation-based features. The slow computational performance of DTW is not an issue in this case as each subject only has 90 time series at most. However, DTW's performance issue will be apparent for an fMRI brain image segmentation study as a large number of time series needs to be computed (more than 150,000 voxels).

2.3.3 Deep learning

In [29], Sarraf and Tofighi proposed a deep 2D CNN to detect Alzheimer disease, in which the 4D fMRI signal is decomposed into several 2D images. A brain slice in each time step is considered as an image pattern. This strategy increases the variants of data and the size of the dataset. Their proposed network is a variant of LeNet-5 architecture.

The network was trained for 30 epochs using a batch size of 64. The authors used an initial learning rate of 0.01 on a learning rate decay of 0.1 for every 10 epochs. However, this architecture does not consider the temporal nature of fMRI data.

Recently, Riaz *et al.* developed a deep learning method, namely the deep fMRI, to detect ADHD using fMRI [7]. In their dataset, the brain is segmented into 90 regions, and each segmented region is represented by a time-series signal. A training/test sample consists of time series from the 90 regions. The deep fMRI was trained end-to-end with the cross-entropy loss function.

Deep fMRI is able to capture the temporal information of fMRI signal with its

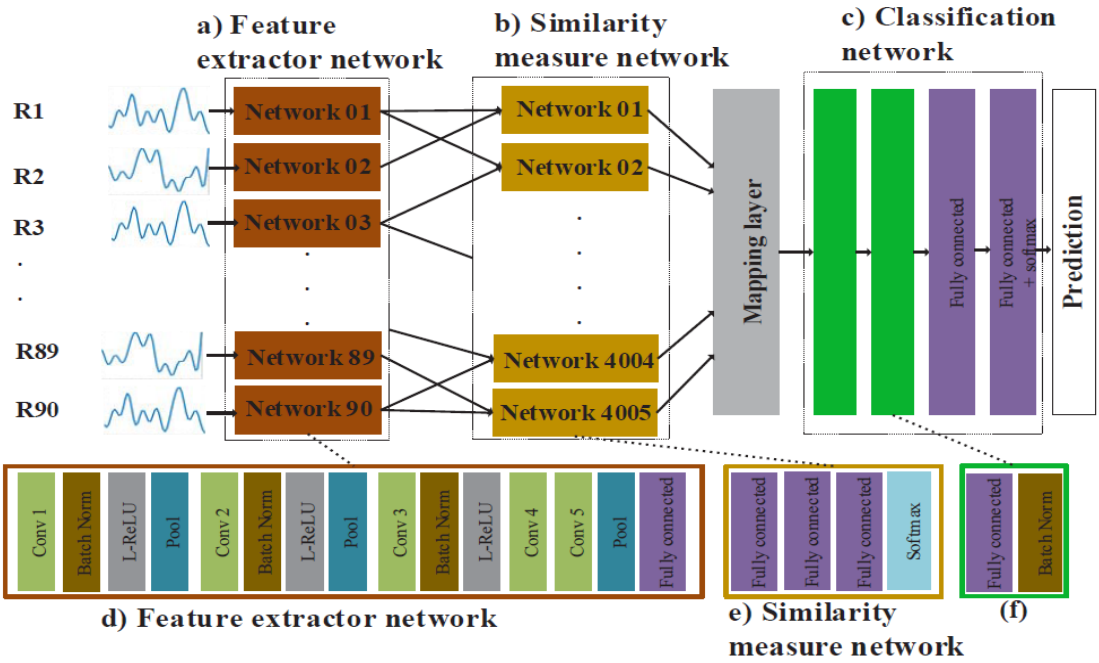


Figure 2.7: The deep fMRI architecture [7].

complex design. Deep fMRI has three components: a feature extractor network, a similarity network, and a classification network. The feature extractor network is implemented with 1D CNNs, where features are extracted from each of the 1D time series. The similarity measure network is implemented with Siamese-inspired neural networks. It learns to measure the similarity between pairs of extracted features from two brain regions. The output is a similarity score. All the outputs from the similarity measure network are fed to a mapping layer. Lastly, the features are passed to the classification network, which is implemented with *softmax*. Riaz *et al.*'s experiments showed that the deep fMRI outperformed the current state-of-the-art of ADHD classification.

2.3.4 fMRI brain tissue segmentation

To the best of our knowledge, there are to date no published papers on fMRI brain tissue segmentation except two preliminary works [35, 36]. In [35], Hutten developed a semi-automatic MATLAB-based GUI segmentation method that utilizes a threshold segmentation technique. Let q be the feature extracted from the raw

fMRI image. Let g be the output. The rule of classification is defined as

$$g(x,y) = \begin{cases} 0, & \text{if } q(x,y) < \theta, \\ 1, & \text{if } q(x,y) \geq \theta, \end{cases} \tag{2.8}$$

where x and y are the coordinates. It requires a threshold θ for each feature that is set manually by the user. Features extracted are: temporal mean, temporal standard deviation, element wise division of standard deviation by mean, standard deviation of spatially smoothed data, correlation coefficient, spatial standard deviation, and spatial mean.

Using similar features, Tan developed a Bayesian classifier that uses a normalized histogram as the probability density function (pdf) of features [36]. Tan experimented with 1D (e.g. mean only) and 2D (e.g. mean and standard deviation) histogram. The experiments were performed as binary classification for each tissue type (e.g. class 1: gray matter and class 2: non-gray).

Bayesian decision theory is based on calculating the consequences between various classification decisions using probability and accompanying costs [37]. Let $P(\omega_j|x)$ denote the *posterior probability* for a class ω_j given feature x. Let $P(\omega_j)$ denote the *prior probability*. The Bayesian formula is defined as [37]

$$P(\omega_j|x) = \frac{p(x|\omega_j)\ P(\omega_j)}{p(x)}, \tag{2.9}$$

where $p(x|\omega_j)$ is the *probability density function* that determines the likelihood that a sample from class ω_j has a feature x, and $p(x)$ is the *evidence* that scales the posterior probability sum to one.

In two-category classification, we can decide x is of class ω_1 if

$$(\lambda_{21} - \lambda_{11})\ P(\omega_1|x) > (\lambda_{12} - \lambda_{22})\ P(\omega_2|x), \tag{2.10}$$

where λ_{ij} is the cost of classifying ω_i when the true class is ω_j. By using the

Bayesian formula (2.9), it can be rewritten as

$$\frac{p(x|\omega_1)}{p(x|\omega_1)} > \frac{(\lambda_{12} - \lambda_{22})}{(\lambda_{21} - \lambda_{11})} \frac{P(\omega_2)}{P(\omega_1)}. \tag{2.11}$$

It can be further simplified into

$$\frac{p(x|\omega_1)}{p(x|\omega_2)} > \theta, \tag{2.12}$$

where θ is the threshold.

The probability density function used in Tan's work is acquired using a histogram technique. The histogram is normalized with

$$p(x|\omega_i) = \frac{h_i(x)}{\sum h_i(x)}, \tag{2.13}$$

where h_i denotes the histogram for class i. The investigations by Tan showed that a 2D pdf outperforms a 1D pdf.

Experiments performed by [35] and [36] suggest that fMRI time series contains useful information for classifying brain tissue types. However, they have not explored this direction fully yet, as only simple statistical features (such as mean and standard deviation) were used.

2.4 Chapter summary

This chapter presented a brief introduction to MRI and fMRI, and reviewed existing works on MRI brain tissue segmentation and fMRI signal classification. Deep learning techniques have been widely applied for MRI brain tissue segmentation. Current deep learning methods can be categorized as a patch-wise or a semantic approach. The advantages and disadvantages of both approaches are summarized as follows:

1. A patch-wise approach has a higher computation time than a semantic ap-

proach. This is because a patch-wise approach has redundant computation as the patches do overlap. In contrast, a semantic approach processes all the pixels in one forward pass.

2. The effectiveness of a patch-wise approach relies on the patch size selected as it determines the information made available to the network. A semantic approach takes the entire MRI image as input, which provides contextual information of the whole image to the network.

3. A semantic approach requires more data to train. In a patch-wise approach, many patches can be extracted from an individual MRI image.

4. Processing 3D or 4D inputs are challenging for a semantic approach as this would result in a high memory requirement.

Recent fMRI research mainly focuses on fMRI signal classification, such as classifying the brain state of a subject and detecting diseases. Most methods use a feature-based approach due to the high dimensionality of fMRI data. Recently, deep learning, where the feature extraction is automated, has become more and more popular.

There are very few papers on fMRI brain tissue segmentation. Current fMRI classification methods are not suitable for brain tissue segmentation because they do not model the spatio-temporal information of fMRI data. Moreover, MRI brain tissue segmentation methods discussed have the following shortcomings if directly applied to an fMRI brain tissue segmentation study:

1. The network architectures discussed focus on spatial domain only. Thus, the accuracy might be poorer because the spatial resolution of fMRI images is lower than MRI images (see Figure 1.2).

2. Most of the network architectures discussed contain a large number of trainable parameters. This adds to the difficulty of training the network (slow convergence, longer training time). To prevent overfitting, a large dataset is

required to tune the hyperparameters. Moreover, the lower spatial resolution of fMRI data suggests that a simpler network is sufficient.

3. Temporal information of fMRI data is lost as none of the methods discussed can model the temporal information.

From the above studies, the promising directions for fMRI brain tissue segmentation research are listed as follows:

1. Develop an automatic fMRI brain tissue segmentation method. In this research, a patch-wise segmentation method based on deep learning is proposed.

2. Explore how temporal information of fMRI data can be used to inform tissue classification. In this thesis, experiments and analysis are conducted to investigate the importance of temporal, spatial and spatio-temporal information.

Chapter 3

Functional MRI Data Acquisition and Preprocessing

Chapter contents

This chapter describes the fMRI dataset used in this research, including the data acquisition and preprocessing. The chapter is organized as follows. Section 3.1 describes the fMRI data, including the machine and acquisition method. Section 3.2 explains the dataset format. Section 3.3 presents the preprocesing steps for the fMRI data.

3.1 MRI machine and data acquisition

The fMRI data used in this thesis were contributed by Puckett *et al.* [8]. The data were scanned at the Centre for Advanced Imaging, University of Queensland, and were acquired using a Siemens MAGNETOM TIM Trio 3T MRI scanner with a 32-channel head coil.

Figure 3.1: The MAGNETOM Trio 3T MRI scanner.

The fMRI images were acquired using a gradient-echo sequence with a field of view (FOV) of 192 mm × 192 mm and an isotropic resolution of 0.8 mm, which produced an image size of 240 × 240 pixels. A total of 37 slices were acquired with a repetition time (TR) of 4 s, a flip angle of 90 degrees, and an echo time (TE) of 38 ms. Subjects were shown an expanding-ring stimulus (see Figure 3.2) that lasted 3 minutes and 4 seconds for each run. All subjects had no visual impairments and no history of psychiatric diseases.

With the above configurations, an fMRI run (a continuous measurement) has 46 volumes, i.e. one volume is obtained every 4 s. The size of an fMRI run is $T \times w \times h \times z$, where T is the number of time steps, w is the image width, h is the image height, and z is the total number of slices (see Figure 1.1). In our case, $T = 46$, $w = 240$, $h = 240$, and $z = 37$.

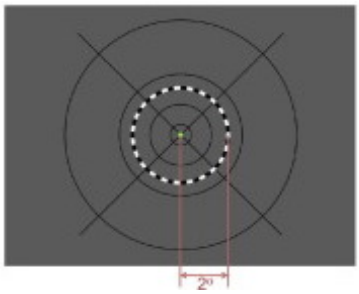


Figure 3.2: Ring stimuli used in [8].

Figure 3.3: All 37 slices in a brain volume.

3.2 fMRI dataset

To provide the ground-truth, we manually annotated the brain voxels using the ITK-SNAP software [38]. The classes annotated were: left gray matter, left white matter, right gray matter, right white matter, blood vessel, non-brain, cerebrospinal fluid and cerebellum. The left and right hemispheres of the brain were labelled differently to broaden the re-usability of our fMRI dataset. Voxels that

were not annotated are considered as non-brain voxels. Additionally, binary brain masks were created to reduce the amount of non-brain voxels.

Two subjects were selected out of Pucket *et al.*'s dataset, and one fMRI run was selected from each subject. Some slices were discarded due to the motion-correction artefact (the first slice) or labelling uncertainty (the last seven slices). The number of voxels for each tissue type is summarized in Table 3.1. Each voxel is represented as a time series with 46 points.

Table 3.1: Summary of the fMRI dataset.

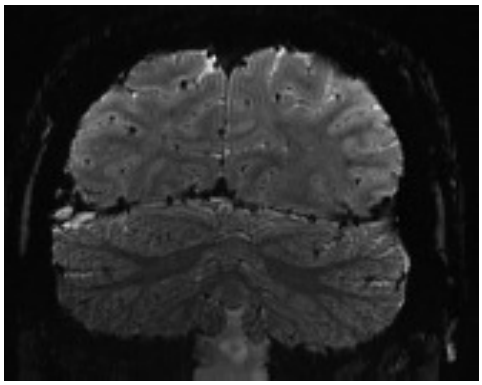
Class	**Samples (voxels)**
Gray matter (GM)	144,198
White matter (WM)	92,795
Blood vessel (BV)	5,905
Non-brain (NB)	54,838
Cerebrospinal fluid (CSF)	18,425
Total	316,161

3.3 Preprocessing of fMRI data

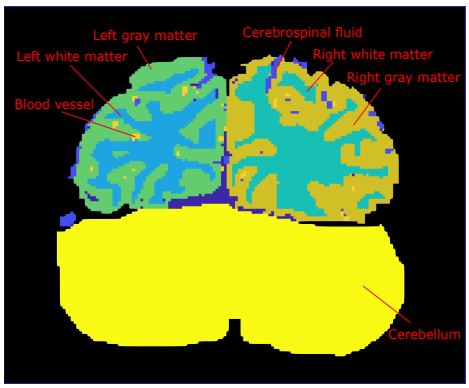
Several preprocessing steps were performed on the fMRI data to remove unwanted and known confounds. One example is motion correction, which reduces the effects of subjects' movements between repeated fMRI-volume scanning. For this thesis, motion correction, slice scan time correction and intensity normalization were performed.

3.3.1 Motion correction

Although the subject is instructed not to move during scanning, involuntary movement is inevitable and must be taken into account. Even if the motion is small, it can corrupt the raw BOLD responses to the extent that changes of intensity between frames are reflected by not only the changes in cerebral physiology [39].



(a) The raw fMRI mean image.



(b) The ground-truth image.

(c) The binary brain mask.

Figure 3.4: Example of a raw fMRI image with the corresponding ground-truth and binary brain mask.

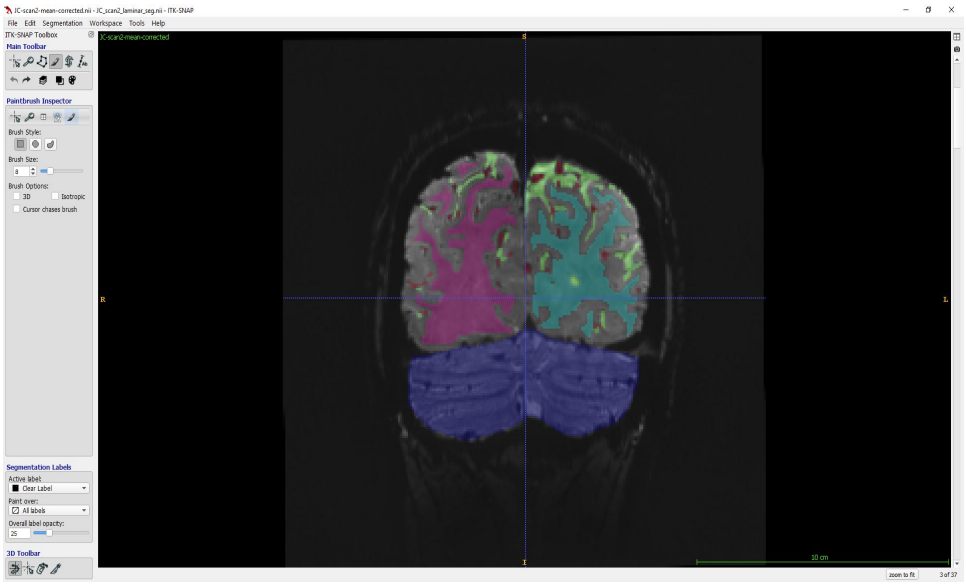


Figure 3.5: A screenshot of the ITK-SNAP software. Ground-truth is annotated manually voxel-by-voxel.

This effect can be reduced with motion correction. Motion correction estimates the body movement parameters, and realigns the time series of brain images using 6 parameters of rigid body movement (3 rotations and 3 translations). The data were motion corrected using the SPM 8 software by Puckett *et al.* [8].

3.3.2 Slice scan time correction

The fMRI brain volume (3D) is scanned slice-by-slice (2D images), and stacking the slices in the order they are taken will form the complete 3D brain volume. Here, each slice acquisition takes 4 seconds and will have a small acquisition delay, which adds up to a significant temporal shifts. Hence, we can not treat the brain volume as a single time instance.

Slice scan time correction (STC) addresses this issue. STC temporally aligns individual slice to a reference slice based on its relative timing using a resampling method [40]. The data were slice scan time corrected using the SPM 8 software by Puckett *et al.* [8].

3.3.3 Intensity normalization

The image intensities recorded in MRI images are unitless. Their numerical values are essentially arbitrary, and only the relation of values between different voxels convey a meaningful relation of signal strength. Different MRI scanners and acquisition settings will yield different intensity ranges.

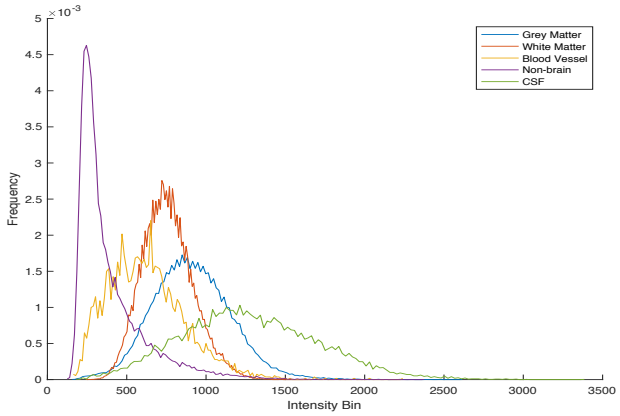
To propose a method that has high compatibility, we need to normalize the data so they fall under the same intensity range. Furthermore, machine learning algorithm works better with normalized data. For example in [41], the K-means clustering algorithm produced a more accurate and efficient result after using a normalized dataset.

Here, z-score formula was used to normalize the data. The mean μ and standard deviation σ were calculated from the voxels in the training set. The

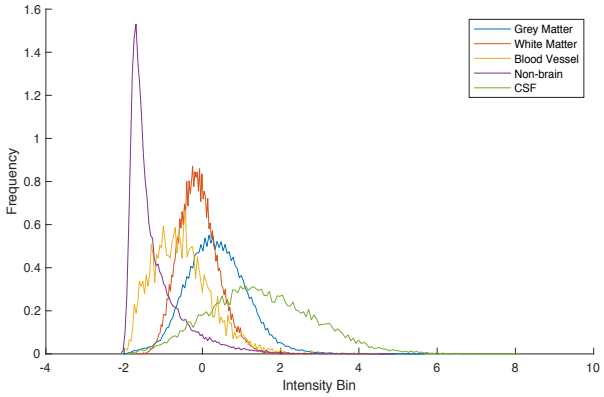
voxel intensity z_i of the i-th voxel is normalized with

$$z_i = \frac{x_i - \mu_{train}}{\sigma_{train}}. \tag{3.1}$$

The effect of the z-score normalization is shown in Figure 3.6.



(a) Before z-score normalization.



(b) After z-score normalization.

Figure 3.6: The histograms of intensity distribution illustrate the effect of the z-score intensity normalization. The scale of the intensity bin is in the range of $[-3, 8]$ after normalization.

3.4 Chapter summary

The fMRI data were acquired using a gradient-echo sequence, which allows rapid scanning of the brain. With the acquisition configurations, the dimensionality of an fMRI run is $46 \times 240 \times 240 \times 37$ voxels. Only necessary preprocessing steps

were performed on the fMRI data to maximize the compatibility of the proposed method. The preprocessing steps performed were motion correction, slice scan time correction and intensity normalization.

Chapter 4

Proposed Method

Chapter contents

In this chapter, we propose a patch-wise segmentation approach for human brain tissue segmentation in fMRI. The proposed approach aims to classify an fMRI voxel into five classes: gray matter (GM), white matter (WM), blood vessel (BV), non-brain (NB), and cerebrospinal fluid (CSF).

The proposed method uses a long-term recurrent convolutional network (LRCN), which is able to utilize the spatio-temporal information of fMRI data. It comprises two state-of-the-art components: convolutional neural network (CNN) for learning spatial features, and long short-term memory (LSTM) for learning temporal features. The components were selected based on their performance on their own

as explored and identified in Chapter 5. The deep learning architecture is inspired by Donahue *et al.* [42].

The LRCN contains three sequential stages: i) spatial feature extraction stage, ii) temporal feature extraction stage, and iii) output stage (see Figure 4.1). These three stages are presented in Section 4.1, 4.2 and 4.3, respectively. The training algorithm for the proposed model is described in Section 4.4.

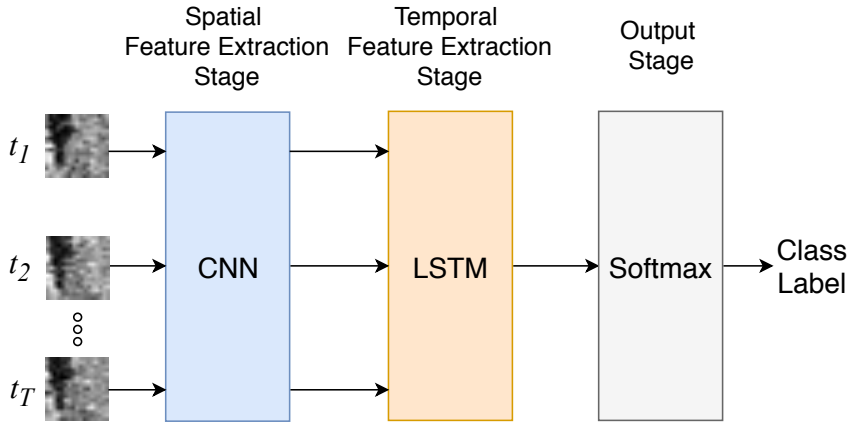
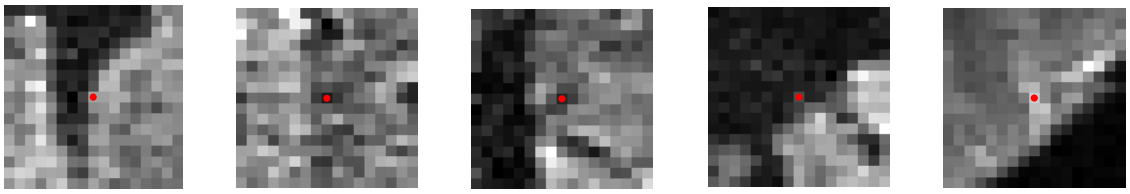


Figure 4.1: Block diagram of the proposed network.

4.1 Spatial feature extraction stage

The inputs for this stage are multiple 2D patches (see Figure 4.2), which are centered at the voxel to be classified and are recorded at different time steps. A CNN based on LeNet-5 [43] is trained to extract features from the input patches. The CNN consists of convolutional layers, max pooling layers, and an output layer, as shown in Figure 4.3.



(a) GM patch (b) WM patch (c) BV patch (d) NB patch (e) CSF patch

Figure 4.2: Example of input (patch size of 17×17 pixels) for each tissue type. The red dot indicates the voxel of interest.

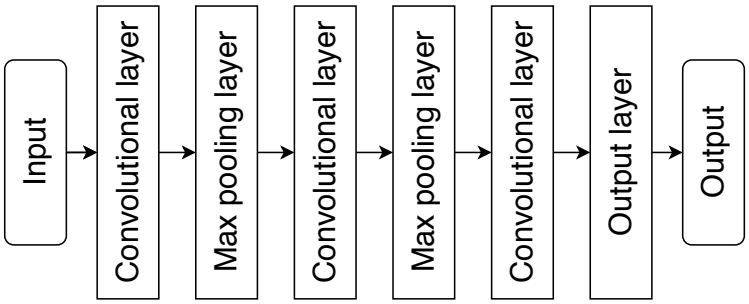


Figure 4.3: The CNN used in the spatial feature extraction stage.

4.1.1 Convolutional layer of CNN

A convolutional layer consists of several filters, each of which is connected to a number of feature maps of the preceding layer. The filter is represented by a matrix of adjustable weights (i.e. a convolution kernel). During training, each filter learns to extract relevant features from its input. The 2D output of a filter is called a *feature map* because it indicates the presence of a feature in the filter's input. Let W_n^l be the n-th filter in layer l and g be a non-linear activation function. The feature map y_n^l is computed as

$$y_n^l = g(\sum_{m \in s} y_m^{l-1} \otimes W_n^l + b_n^l), \tag{4.1}$$

where s is the list of previous layer's feature maps that are connected to filter W_n^l, b_n^l is a bias term, and $\otimes$ is the convolution operator.

Suppose that $h \times w$ is the size of input feature map y_m^{l-1}, the size of the output feature map is

$$D \times E = (h - d + 1) \times (w - e + 1), \tag{4.2}$$

where $d \times e$ is the size of convolutional filter W_n^l. The output size of the entire layer is then

$$D \times E \times N, \tag{4.3}$$

where N is the total number of filters in this layer.

4.1.2 Max pooling layer of CNN

A max pooling layer reduces the size of its input feature maps by a fixed factor. Each neuron in this layer has a one-to-one connection to the preceding filter, thus the number of neurons in this layer must match the number of filters in the previous layer.

To perform max pooling, the feature map is first partitioned into non-overlapping squares. In each square, the maximum value is taken as the next output. If necessary, the feature map can be padded with zeros to ensure the last row and column are pooled (see Figure 4.4).

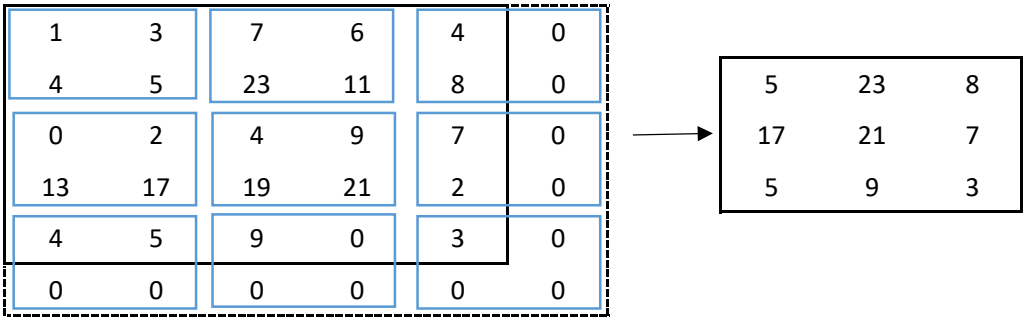


Figure 4.4: An illustration of the max pooling operation.

After the max pooling operation, the size of the feature map becomes

$$D \times E = \left\lceil \frac{D}{2} \right\rceil \times \left\lceil \frac{E}{2} \right\rceil. \tag{4.4}$$

The output size of the entire layer is

$$D \times E \times N, \tag{4.5}$$

where N is the number of filters in the preceding convolutional layer.

4.1.3 Output layer of CNN

The output layer is also known as a fully connected layer, where each neuron is connected to every neuron in the previous layer. The scalar output of the n-th

neuron in output layer L is defined as

$$y_n^L = g(\sum_{m \in s} y_m^{L-1} \times W_{m,n}^L + b_n^L), \tag{4.6}$$

where s is the list of all neurons in layer $L-1$, and $W_{m,n}^L$ is the weight from the m-th neuron in layer $L-1$ to the n-th neuron in layer L.

The outputs of all the neurons in this layer form the CNN output:

$$\mathbf{y} = [y_n^L, y_2^L, \dots]. \tag{4.7}$$

where N is the total number of neurons in this layer.

4.2 Temporal feature extraction stage

This stage applies a long short-term memory to process the outputs of the CNN. LSTM is a recurrent neural network (RNN) that can solve the vanishing and exploding gradient problems faced by the traditional RNN [44]. We adopt the LSTM version proposed by Graves [45].

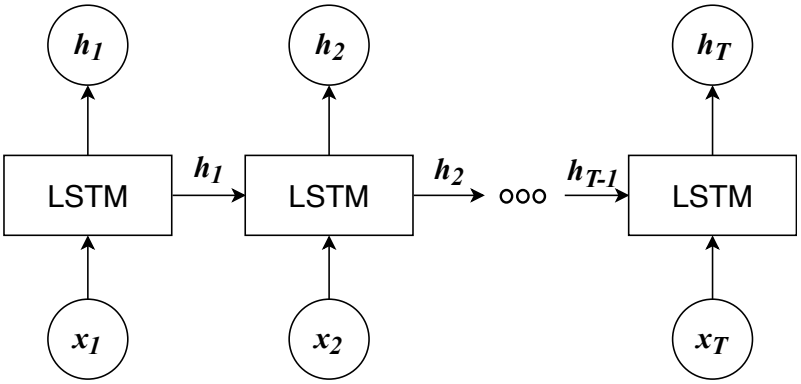


Figure 4.5: Illustration of the LSTM's recurrent characteristic.

Let $\mathbf{x}_t$ be the input to the LSTM at time step t, that is, $\mathbf{x}_t = \mathbf{y}_t$ produced by the CNN. The LSTM consists of an input gate $\mathbf{i}_t$, a forget gate $\mathbf{f}_t$, an output gate $\mathbf{o}_t$, a cell state $\mathbf{c}_t$, and a hidden state $\mathbf{h}_t$ at each time step t. The operation of each gate is described next (refer also to Figure 4.5).

The input gate is defined as

$$\mathbf{i}_t = \sigma(W_{x,i}\,\mathbf{x}_t + W_{h,i}\,\mathbf{h}_{t-1} + W_{c,i}\,\mathbf{c}_{t-1} + b_i), \tag{4.8}$$

where b_i is the bias for the input gate, $W_{x,i}$ is the weight from the input to the input gate, $W_{h,i}$ is the weight from the hidden state to the input gate, and $W_{c,i}$ is the weight from the cell state to the input gate. Here, σ is the sigmoid function.

The forget gate is defined as

$$\mathbf{f}_t = \sigma(W_{x,f}\,\mathbf{x}_t + W_{h,f}\,\mathbf{h}_{t-1} + W_{c,f}\,\mathbf{c}_{t-1} + b_f), \tag{4.9}$$

where b_f is the bias for the forget gate, $W_{x,f}$ is the weight from the input to the forget gate, $W_{h,f}$ is the weight from the hidden state to the forget gate, and $W_{c,f}$ is the weight from the cell state to the forget gate.

The output gate is defined as

$$\mathbf{o}_t = \sigma(W_{x,o}\,\mathbf{x}_t + W_{h,o}\,\mathbf{h}_{t-1} + W_{c,o}\,\mathbf{c}_t + b_o), \tag{4.10}$$

where b_o is the bias for the output gate, $W_{x,o}$ is the weight from the input to the output gate, $W_{h,o}$ is the weight from the hidden state to the output gate, and $W_{c,o}$ is the weight from the cell state to the output gate.

The cell state is defined as

$$\mathbf{c}_t = \mathbf{f}_t\,\mathbf{c}_{t-1} + \mathbf{i}_t \tanh(W_{x,c}\,\mathbf{x}_t + W_{h,c}\,\mathbf{h}_{t-1} + b_c), \tag{4.11}$$

where b_c is the bias for the cell state, $W_{x,c}$ is the weight from the input to the cell state, and $W_{h,c}$ is the weight from the hidden state to the cell state. The input gate $\mathbf{i}_t$ determines what to insert, and the forget gate $\mathbf{f}_t$ determines what to remove from the cell state at time step t.

The hidden state is defined as

$$\mathbf{h}_t = \mathbf{o}_t \ \tanh(\mathbf{c}_t). \tag{4.12}$$

The hidden state $\mathbf{h}_t$ is the output vector of the LSTM at time step t. The size of the hidden state is a tunable parameter.

4.3 Output stage

The output stage uses the last hidden state $\mathbf{h}_T$ of the LSTM as input. It applies a *softmax* classifier to predict the brain tissue class for the voxel of interest. Let C be the number of total classes, $C = 5$ in our case. The *softmax* classifier consists of C fully connected neurons. Each neuron first computes a weighted sum of its inputs

$$z_c = W_c^\top \ \mathbf{h}_T + b_c, \tag{4.13}$$

where W_c is the weight vector of the neuron, $c = 1, 2, \ldots, C$.

The neuron then produces an output as

$$p_c = \frac{\exp(z_c)}{\sum_{n=1}^{C} \exp(z_n)}. \tag{4.14}$$

The outputs of all neurons $\mathbf{p} = [p_1, p_2, \ldots, p_C]$ are interpreted as the predicted class probabilities for the input voxel.

4.4 Training algorithm

The objective function used for training the proposed network is the categorical cross-entropy function:

$$L = -\sum_{k=1}^{K} \hat{\mathbf{y}}_k^\top \log(\mathbf{y}_k), \tag{4.15}$$

where $\mathbf{y}_k$ is the network output vector for the k-th input sample, $\hat{\mathbf{y}}_k$ is the corresponding desired output vector, and K is the number of training samples.

Optimization is performed using the Adam algorithm proposed in [46]. This is a first-order gradient-based algorithm with several attractive properties. It is computationally efficient, has low memory requirements, is suitable for problems with noisy gradients, and requires little tuning. The Adam optimizer is described as follows.

Given the gradient g_t at time t, the first moment estimate m_t and second raw moment estimate v_t are defined as

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t, \\ v_t &= \beta_2 v_{t-1} + (1-\beta_2) g_t^2, \end{aligned} \tag{4.16}$$

where β_1 and β_2 are two exponential decay rates. Then, m_t and v_t are bias corrected with

$$\begin{aligned} \hat{m}_t &= \frac{m_t}{1-\beta_1^t}, \\ \hat{v}_t &= \frac{v_t}{1-\beta_2^t}. \end{aligned} \tag{4.17}$$

Finally, the network parameter θ_t is updated with the following rule:

$$\theta_t = \theta_{t-1} - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}, \tag{4.18}$$

where ϵ is a tunable parameter and α is the learning rate. The value used for the parameters are: $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and $\alpha = 0.0001$. The parameters' value chosen are the recommended default settings [46], except α is tuned manually.

To prevent overfitting, the dropout strategy as described in [47] is applied to the CNN's output layer and the LSTM layer. During training, each neuron in the layer where the dropout mechanism is applied, has a probability of getting disconnected.

4.5 Chapter summary

A novel deep learning method is proposed for fMRI brain tissue segmentation. The proposed method consists of three stages. The first stage uses a CNN to extract spatial features, the second stage uses a LSTM to extract temporal features, and the third stage uses a *softmax* classifier to predict the brain tissue class. The proposed model is trained end-to-end using the Adam optimizer with the categorical cross-entropy loss function.

Chapter 5

Experiments and Analysis

Chapter contents

In this chapter, we implemented and evaluated several classifiers that accept inputs from three sources: temporal, spatial, and spatio-temporal domains. The chapter is organized as follows. Section 5.1 describes the experimental methods. Section 5.2 presents the hyperparameters of the proposed method. Section 5.3 and 5.4 analyze the temporal domain and spatial domain classifiers, respectively. Section 5.5 presents the result of the proposed method and compares it with other methods.

5.1 Experimental methods

The evaluation was conducted using five-fold cross-validation. The slices of each subject were first shuffled with random permutations. The collected voxels were then divided into five approximately equal partitions in terms of subjects, slices, and tissue types. For each fold, one partition was used as the test set, and the remaining partitions were used as the training set. This process was repeated five times for different choices of the test set. Note that each training set was further divided into 80% samples for training, and 20% samples for validation. The validation samples were used for early stopping in neural network training.

To ensure a fair comparison, each classifier experimented here used its respective optimum hyperparameters. The neural networks were implemented using the Keras library with Tensorflow backend [48]. During training, the weights were saved only if the validation accuracy had improved from the previous epoch. Training was stopped early if the validation accuracy had not improved after 10 epochs. Other classifiers were implemented using MATLAB.

The classifiers were evaluated using three methods: confusion matrix, overall classification rate and Dice similarity coefficient. The confusion matrix was constructed using the averaged classification rate of the test set in all five folds. The columns of the confusion matrix represent the true class whereas the rows

represent the predicted class.

The overall classification rate (CR) is the percentage of the test samples that are correctly classified. Let C be the number of classes, $C = 5$. The overall classification rate is computed as

$$CR = \frac{1}{N}\sum_{c=1}^{C} TP_c\,, \tag{5.1}$$

where N is the total samples, and TP denotes the true positive. The CR measure was used as the main criterion for training the classifiers and selecting the optimum hyperparameters for the classifiers.

The Dice similarity coefficient (DSC) is the harmonic mean of precision and recall. In this research, per class DSC and average DSC were computed. The DSC for class c is calculated as

$$DSC_c = \frac{2TP_c}{2TP_c + FP_c + FN_c}, \tag{5.2}$$

where FP is the false positive, and FN is the false negative. The average DSC is defined as

$$DSC_{avg} = \frac{1}{C}\sum_{c=1}^{C} DSC_c\,. \tag{5.3}$$

A higher DSC means a more accurate classifier.

5.2 Hyperparameters of the proposed method

From the literature review, the popular patch sizes are 13×13, 17×17, 25×25, 29×29, 51×51 and 75×75 pixels. We chose patch size of 17×17 pixels for the proposed method because of three reasons. First, patch size of 17×17 pixels is a reasonable trade-off between memory requirement and spatial information. Second, Zhang *et al.*'s experiments showed that patch size of 17×17 pixels performs well in MRI brain tissue segmentation problem [2]. Third, patch with lower spatial

information is used to observe the significance of temporal information.

The proposed network has a large number of hyperparameters to optimize, such as the number of filters and the filter's size in each convolutional layer. To optimize the hyperparameters, Hyperopt library [49] with Tree of Parzen Estimator algorithm was used to search the defined parameter spaces. For this purpose, the validation samples of fold one were split into 80% for training and 20% for testing.

The hyperparameter search was conducted using the following parameters: *learning rate* = 0.0001, *max epoch* = 50, and *batch size* = 64. A total of 40 trials were performed. The best hyperparameters are given in Table 5.1.

Table 5.1: The optimum hyperparameters for the proposed method.

Stage	**Layer**	**Settings**
Spatial feature extraction: CNN	Convolutional	128 filters of size 5 x 5
	Max pooling	max pool size of 2 x 2
	Convolutional	32 filters of size 3x3
	Max pooling	max pool size of 2x2
	Convolutional	16 filters of size 3x3
	Output	16 neurons
	Dropout	27%
Temporal feature extraction: LSTM	LSTM	hidden vector size of 64
	Dropout	46%
Output: Softmax	Softmax	5 neurons

5.3 Analysis of temporal domain classifiers

In this section, four temporal classifiers were implemented: Bayesian classifier, k-nearest neighbor (k-NN) classifier, LSTM classifier, and LSTM-FCN classifier.

The Bayesian and k-NN classifiers use two features, which are the mean and the standard deviation (std) of each time series of the voxel of interest. The LSTM and LSTM-FCN classifiers use the time series of the voxel directly.

5.3.1 Bayesian classifier

We extended the Bayesian classifier described in Section 2.3.4 into a multi-class classifier. Instead of the decision rule defined in (2.12), here, feature $\mathbf{x}$ is assigned to class ω_i if $p(\mathbf{x}|w_i) > p(\mathbf{x}|w_j)$ for all $j \neq i$. The class-conditional pdfs were estimated as normalized 2D histograms of mean and std features.

The optimum histogram bin sizes for both features were selected via a grid search. A total of 25 combinations of bin sizes were explored, and the results are presented in Table 5.2. Bin size of 30 for feature 1 (mean) and bin size of 10 for feature 2 (std) were selected, as they resulted in the highest accuracy.

Table 5.2: Grid search for finding the optimum histogram bin sizes for the Bayesian classifier.

		Feature 1: Mean				
		10	20	30	40	50
Feature 2: Std	10	0.3866	0.5241	**0.5498**	0.5069	0.5147
	20	0.3517	0.4884	0.5105	0.5092	0.5059
	30	0.3771	0.5054	0.5221	0.5277	0.5255
	40	0.3783	0.5079	0.5057	0.5117	0.5152
	50	0.3627	0.5012	0.5106	0.5024	0.5063

Table 5.3: The confusion matrix and DSCs of the Bayesian classifier.

		True Class				
		GM	WM	BV	NB	CSF
Predicted Class	GM	**36%**	19%	25%	5%	25%
	WM	42%	**78%**	46%	15%	14%
	BV	0%	0%	**0%**	0%	0%
	NB	5%	1%	15%	**71%**	5%
	CSF	16%	2%	14%	8%	**56%**
	DSC	46.81%	60.35%	0%	74.81%	34.69%

Table 5.3 presents the confusion matrix and DSCs of the Bayesian classifier. The confusion matrix shows classification rates of 36% for GM, 78% for WM, 0% for BV, 71% for NB, and 56% for CSF. For GM samples, the classification rate was

low (36%). Here, 42%, 5% and 16% of the GM samples were misclassified as WM, NB and CSF, respectively. For WM samples, the classification rate was the highest (78%). In this experiment, 19%, 1% and 2% of the WM samples were misclassified as GM, NB and CSF, respectively.

For BV samples, the classification rate was the lowest (0%). Here, 25%, 46%, 15% and 14% of the BV samples were misclassified as GM, WM, NB and CSF, respectively. For NB samples, the classification rate was the second highest (71%). In this evaluation, 5%, 15% and 8% of the NB samples were misclassified as GM, WM and CSF, respectively. For CSF samples, the classification rate was poor (56%). Overall, 25%, 14% and 5% of the CSF samples were misclassified as GM, WM and NB, respectively.

5.3.2 k-NN classifier

This classifier identifies, for a given test sample, the k nearest neighbors from the training set. It then uses majority voting on the neighbors' labels to determine the label of the test sample. The Euclidean distance metric was used in this experiment. The number of neighbors $k = 201$ was selected via an exhaustive search.

Table 5.4: The confusion matrix and DSCs of the temporal k-NN classifier.

		True Class				
		GM	WM	BV	NB	CSF
Predicted Class	GM	**67%**	41%	53%	17%	64%
	WM	26%	**58%**	31%	11%	9%
	BV	0%	0%	**0%**	0%	0%
	NB	5%	1%	14%	**70%**	4%
	CSF	1%	0%	3%	2%	**22%**
	DSC	64.12%	55.86%	0%	74.89%	32.01%

Table 5.4 presents the confusion matrix and DSCs of the k-NN classifier. The confusion matrix shows classification rates of 67% for GM, 58% for WM, 0% for

BV, 70% for NB, and 22% for CSF. For GM samples, the classification rate was the second highest (67%). Here, 26%, 5% and 1% of the GM samples were misclassified as WM, NB and CSF, respectively. For WM samples, the classification rate was low (58%). In this experiment, 41% and 1% of the WM samples were misclassified as GM and NB, respectively.

For BV samples, the classification rate was the lowest (0%). Here, 53%, 31%, 14% and 3% of the BV samples were misclassified as GM, WM, NB and CSF, respectively. For NB samples, the classification rate was the highest (70%). In this evaluation, 17%, 11% and 2% of the NB samples were misclassified as GM, WM and CSF, respectively. For CSF samples, the classification rate was low (22%). Overall, 64%, 9% and 4% of the CSF samples were misclassified as GM, WM and NB, respectively.

5.3.3 LSTM classifier

This classifier has the same architecture as the proposed network, but without the spatial feature extraction stage (see Table 5.1). The LSTM was trained using the Adam optimizer with the categorical cross-entropy loss function. The training parameters were *learning rate* = 0.0001, *max epoch* = 100, and *batch size* = 64.

Table 5.5: The confusion matrix and DSCs of the LSTM classifier.

		True Class				
		GM	WM	BV	NB	CSF
Predicted Class	GM	**71%**	39%	23%	6%	59%
	WM	25%	**59%**	43%	12%	5%
	BV	0%	0%	**0%**	0%	0%
	NB	2%	2%	34%	**82%**	1%
	CSF	2%	0%	0%	0%	**35%**
	DSC	68.52%	56.75%	0%	83.81%	47.07%

Table 5.5 presents the confusion matrix and DSCs of the LSTM classifier. The confusion matrix shows classification rates of 71% for GM, 59% for WM, 0%

for BV, 82% for NB, and 35% for CSF. For GM samples, the classification rate was the second highest (71%). In this evaluation, 25%, 2% and 2% of the GM samples were misclassified as WM, NB and CSF, respectively. For WM samples, the classification rate was poor (59%). Here, 39% and 2% of the WM samples were misclassified as GM and NB, respectively.

For BV samples, the classification rate was the lowest (0%). In this experiment, 23%, 43% and 34% of the BV samples were misclassified as GM, WM and NB, respectively. For NB samples, the classification rate was the highest (82%). Here, 6% and 12% of the NB samples were misclassified as GM and WM, respectively. For CSF samples, the classification rate was low (35%). Overall, 59%, 5% and 1% of the CSF samples were misclassified as GM, WM and NB, respectively.

5.3.4 LSTM-FCN classifier

This classifier, proposed by Karim *et al.* [9], is the current state-of-the-art for time series classification. The LSTM-FCN was trained using the Adam optimizer with the categorical cross-entropy loss function. The training parameters were *learning rate* = 0.001, *max epoch* = 100, and *batch size* = 64.

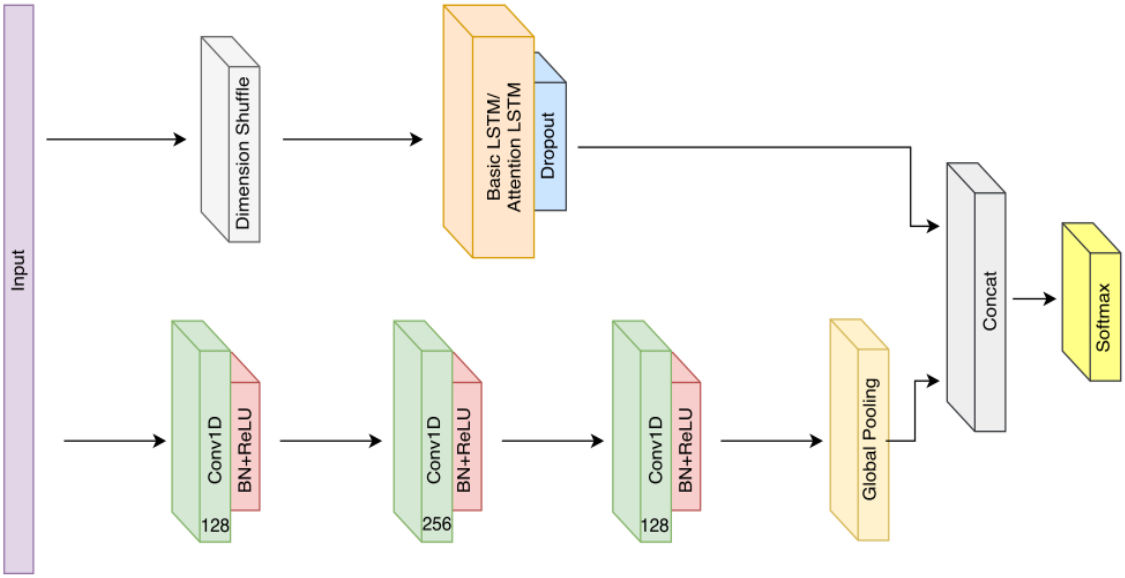


Figure 5.1: The deep LSTM-FCN architecture proposed by [9]. The basic LSTM version is used here.

Table 5.6 presents the confusion matrix and DSCs of the LSTM-FCN classifier. The confusion matrix shows classification rates of 71% for GM, 60% for WM, 0% for BV, 81% for NB, and 35% for CSF. For GM samples, the classification rate was the second highest (71%). Here, 25%, 2% and 1% of the GM samples were misclassified as WM, NB and CSF, respectively. For WM samples, the classification rate was low (60%). In this evaluation, 39% and 1% of the WM samples were misclassified as GM and NB, respectively.

Table 5.6: The confusion matrix and DSCs of the LSTM-FCN classifier.

		True Class				
		GM	WM	BV	NB	CSF
Predicted Class	GM	**71%**	39%	27%	7%	59%
	WM	25%	**60%**	39%	11%	5%
	BV	0%	0%	**0%**	0%	0%
	NB	2%	1%	34%	**81%**	1%
	CSF	1%	0%	0%	0%	**35%**
	DSC	68.44%	57.23%	0%	83.95%	47.19%

For BV samples, the classification rate was the lowest (0%). Here, 27%, 39% and 34% of the BV samples were misclassified as GM, WM and NB, respectively. For NB samples, the classification rate was the highest (81%). In this experiment, 7% and 11% of the NB samples were misclassified as GM and WM, respectively. For CSF samples, the classification rate was poor (35%). Overall, 59%, 5% and 1% of the CSF samples were misclassified as GM, WM and NB, respectively.

5.3.5 Comparison of the temporal domain classifiers

Table 5.7 presents the CR and DSC_{avg} of all the evaluated temporal classifiers. In a descending order, the CRs for the different classifiers were: LSTM-FCN (66.08%), LSTM (65.93%), k-NN (61.44%), and Bayesian (54.94%). This shows that classifiers using machine-learned features (i.e. LSTM and LSTM-FCN) outperformed classifiers using hand-crafted features (i.e. k-NN and Bayesian).

The performance of the shallow LSTM network was very similar to the deep LSTM-FCN network (CR of 65.93% versus 66.08% and DSC_{avg} of 51.12% versus 51.28%). This suggests that a shallow network architecture is sufficient to model the temporal information available in fMRI data.

Table 5.7: The CR and average DSC of the temporal domain algorithms.

Classifier	**CR ± std**	**DSC_{avg} ± std**
Bayesian	54.94% ± 0.46%	43.24% ± 0.45%
k-NN	61.44% ± 1.07%	45.24% ± 1.04%
LSTM	65.93% ± 0.77%	51.12% ± 0.87%
LSTM-FCN	66.08% ± 0.91%	51.28% ± 0.92%

The confusion matrices of the classifiers (Table 5.3, 5.4, 5.5 and 5.6) illustrate that all the temporal algorithms failed to detect any of the BV samples. The best DSC for each tissue class was achieved by the LSTM (68.52%) for GM, Bayesian (60.35%) for WM, LSTM-FCN (83.95%) for NB, and LSTM-FCN (47.19%) for CSF.

Based on this comparison, we conclude that utilizing temporal information in fMRI data alone is insufficient for obtaining good classification performance. The shallow LSTM network was selected as the temporal feature extractor for the proposed method because it had the second highest classification rate, and its performance was very similar to the best classifier in this experiment (deep LSTM-FCN).

5.4 Analysis of spatial domain classifiers

In this section, two spatial classifiers were implemented: k-NN classifier with PCA and deep CNN classifier. The input for these classifiers is a patch of size 17×17 pixels centered on the voxel of interest. In training, patches were taken from every fifth time step to increase the number of samples. In testing, the input to the classifiers was the patch averaged across the 46 time steps.

5.4.1 k-NN classifier with PCA

Principal component analysis is used to extract features, which are then fed into the k-NN classifier. The PCA is applied similarly as [26] (described in Section 2.3.1). Here, two parameters need to be tuned, which are: the number of principal components (PC), and the number of neighbors k.

First, the number of principal components was varied from 1 to 6 to find the optimum. This search used $k = 1$. The result is presented in Table 5.8. The highest accuracy was achieved using two principal components, hence PC = 2 was selected. Then, the optimum k was exhaustive searched using PC = 2. We found that $k = 17$ resulted the best accuracy, hence $k = 17$ was selected.

Table 5.8: Classification rate as a function of the number of principal components.

Principal component	1	2	3	4	5	6
Classification rate	54.97%	**58.63%**	57.72%	56.47%	55.50%	54.52%

Table 5.9: The confusion matrix and DSCs of the spatial k-NN classifier.

		True Class				
		GM	WM	BV	NB	CSF
Predicted Class	GM	**60%**	24%	52%	27%	56%
	WM	31%	**75%**	37%	10%	12%
	BV	0%	0%	**5%**	0%	1%
	NB	6%	1%	3%	**61%**	13%
	CSF	2%	0%	3%	2%	**18%**
	DSC	61.99%	64.16%	7.71%	66.42%	25.67%

Table 5.9 presents the confusion matrix and DSCs of the spatial k-NN classifier. The confusion matrix shows classification rates of 60% for GM, 75% for WM, 5% for BV, 61% for NB, and 18% for CSF. For GM samples, the classification rate was low (60%). Here, 31%, 6% and 2% of the GM samples were misclassified as WM, NB and CSF, respectively. For WM samples, the classification rate was the highest (75%). In this evaluation, 24% and 1% of the WM samples were misclassified as

GM and NB, respectively.

For BV samples, the classification rate was the lowest (5%). Here, 52%, 37%, 3% and 3% of the BV samples were misclassified as GM, WM, NB and CSF, respectively. For NB samples, the classification rate was poor (61%). In this experiment, 27%, 10% and 2% of the NB samples were misclassified as GM, WM and CSF, respectively. For CSF samples, the classification rate was the second lowest (18%). Overall, 56%, 12%, 1% and 13% of the CSF samples were misclassified as GM, WM, BV and NB, respectively.

5.4.2 Deep CNN classifier

This classifier has the same architecture as the proposed network, but without the temporal feature extraction stage (see Table 5.1). The network was trained using the Adam algorithm with the categorical cross-entropy loss function. The training parameters were *learning rate* = 0.0001, *max epoch* = 100, and *batch size* = 64.

Table 5.10: The confusion matrix and DSCs of the deep CNN classifier.

		True Class				
		GM	WM	BV	NB	CSF
Predicted Class	GM	**77%**	8%	36%	9%	40%
	WM	18%	**92%**	6%	0%	1%
	BV	0%	0%	**50%**	0%	1%
	NB	2%	0%	4%	**89%**	4%
	CSF	2%	0%	3%	1%	**54%**
	DSC	80.31%	83.39%	59.69%	90.65%	61.98%

Table 5.10 presents the confusion matrix and DSCs of the deep CNN classifier. The confusion matrix shows classification rates of 77% for GM, 92% for WM, 50% for BV, 89% for NB, and 54% for CSF. For GM samples, the classification rate was the third highest (77%). Here, 18%, 2% and 2% of the GM samples were misclassified as WM, NB and CSF, respectively. For WM samples, the classification

rate was the highest (92%). In this experiment, only 8% of the WM samples were misclassified as GM.

For BV samples, the classification rate was the lowest (50%). Here, 36%, 6%, 4% and 3% of the BV samples were misclassified as GM, WM, NB and CSF, respectively. For NB samples, the classification rate was the second highest (89%). In this evaluation, 9% and 1% of the NB samples were misclassified as GM and CSF, respectively. For CSF samples, the classification rate was poor (54%). Overall, 40%, 1%, 1% and 4% of the CSF samples were misclassified as GM, WM, BV and NB, respectively.

5.4.3 Comparison of the spatial domain classifiers

Table 5.11 presents the CR and DSC_{avg} of all the tested spatial classifiers. The deep CNN outperformed the k-NN in terms of both CR (81.87% versus 61.34%) and DSC_{avg} (75.19% versus 45.17%).

Table 5.11: The CR and average DSC of the spatial domain algorithms.

Classifier	**CR ± std**	**DSC_{avg} ± std**
k-NN	61.34% ± 0.47%	45.17% ± 0.33%
CNN	81.87% ± 1.43%	75.19% ± 1.88%

The confusion matrices of the classifiers (Table 5.9 and 5.10) show that the deep CNN outperformed k-NN at classifying all classes. The DSCs were 80.31% versus 61.99% for GM, 83.39% versus 64.16% for WM, 59.69% versus 7.71% for BV, 90.65% versus 66.42% for NB and 61.98% versus 25.67% for CSF.

Based on this comparison, we conclude that spatial information is crucial in achieving good classification performance in fMRI data. The deep CNN was selected as the spatial feature extractor for the proposed model because it achieved the best performance.

5.5 Analysis of spatio-temporal domain classifier

In this study, the proposed method is the only spatio-temporal classifier. The classifier accepts patches of size 17×17 pixels extracted at 46 time steps as input.

In this section, first, the result of the proposed method is presented and described. Then, comparisons between temporal, spatial, and spatio-temporal classifiers are given.

5.5.1 The proposed method

The hyperparameters of the proposed method are given at Section 5.2 and the methodology is explained in Chapter 4. The training parameters of the proposed method were *max epoch* = 100, *learning rate* = 0.0001, and *batch size* = 64.

Table 5.12: The confusion matrix and DSCs of the proposed method.

		True Class				
		GM	WM	BV	NB	CSF
Predicted Class	GM	**86%**	15%	34%	9%	43%
	WM	9%	**85%**	1%	0%	0%
	BV	1%	0%	**58%**	0%	1%
	NB	2%	0%	4%	**90%**	4%
	CSF	2%	0%	2%	1%	**51%**
	DSC	83.76%	85.46%	63.70%	90.72%	61.28%

Table 5.12 presents the confusion matrix and DSCs of the proposed method. The confusion matrix shows classification rates of 86% for GM, 85% for WM, 58% for BV, 90% for NB, and 51% for CSF. For GM samples, the classification rate was the second highest (86%). Here, 9%, 1%, 2% and 2% of the GM samples were misclassified as WM, BV, NB and CSF, respectively. For WM samples, the classification rate was the third highest (85%). In this evaluation, only 15% of the WM samples were misclassified as GM.

For BV samples, the classification rate was poor (58%). Here, 34%, 1%, 4% and

2% of the BV samples were misclassified as GM, WM, NB and CSF, respectively. For NB samples, the classification rate was the highest (90%). In this experiment, 9% and 1% of the NB samples were misclassified as GM and CSF, respectively. For CSF samples, the classification rate was the lowest (51%). Overall, 43%, 1% and 4% of the CSF samples were misclassified as GM, BV and NB, respectively.

5.5.2 Overall comparison of the explored methods

Table 5.13 presents CR and DSC_{avg} of the evaluated temporal, spatial and spatio-temporal classifiers. In a descending order, the CRs for the classifiers were: proposed method (84.04%), CNN (81.87%), LSTM-FCN (66.08%), LSTM (65.93%), temporal k-NN (61.44%), spatial k-NN (61.34%), and Bayesian (54.94%). It is evident that classifiers using machine-learned features (i.e. the proposed method, CNN, LSTM-FCN and LSTM) outperformed classifiers using hand-crafted features (i.e. temporal k-NN, spatial k-NN and Bayesian).

Table 5.13: The overall CR and DSC of the evaluated classifiers.

Domain	**Classifier**	**CR ± std**	**DSC_{avg} ± std**
Temporal	Bayesian [37]	54.94% ± 0.46%	43.24% ± 0.45%
	k-NN [37]	61.44% ± 1.07%	45.24% ± 1.04%
	LSTM [45]	65.93% ± 0.77%	51.12% ± 0.87%
	LSTM-FCN [9]	66.08% ± 0.91%	51.28% ± 0.92%
Spatial	k-NN [37]	61.34% ± 0.47%	45.17% ± 0.33%
	CNN [43]	81.87% ± 1.43%	75.19% ± 1.88%
Spatio-temporal	Proposed method	84.04% ± 1.32%	76.99% ± 2.02%

The best spatial classifier, namely the CNN, had a CR of 81.87% and a DSC_{avg} of 75.19%. It outperformed the best temporal classifier, namely the LSTM-FCN, with a CR of 66.08% and a DSC_{avg} of 51.28%. The per class DSCs of the CNN versus the LSTM-FCN were 80.31% versus 68.44% for GM, 83.39% versus 57.23% for WM, 59.69% versus 0% for BV, 90.65% versus 83.95% for NB, and 61.98% versus 47.19% for CSF. This result shows that spatial information is useful for detecting the BV samples. BV has a similar intensity characteristic as NB (both classes have low

intensity values in fMRI images). However, BV samples are typically surrounded by GM or WM samples, thus neighboring information helps in identifying BV samples. Overall, the spatial features were more effective than the temporal features in achieving good classification performance.

The proposed method, which utilizes both temporal and spatial information had a CR of 84.04% and a DSC_{avg} of 76.99%, which were higher than those by other classifiers in this study. The per class DSCs of the proposed method versus CNN were 83.76% versus 80.31% for GM, 85.46% versus 83.39% for WM, 63.70% versus 59.69% for BV, 90.72% versus 90.65% for NB, and 61.28% versus 61.98% for CSF. For CSF, the proposed method performed slightly poorer than the CNN, however the difference is insignificant. This demonstrates that temporal information contains useful features that can boost the overall classification performance.

Figure 5.2 depicts the segmentation results of the proposed method. The majority of the incorrect voxels were at the boundary between two tissue classes. Voxels at the boundaries are likely to sample multiple tissue types in various fractions, so the class label can be imprecise.

5.6 Chapter summary

This chapter presented and analyzed the experimental results of the temporal, spatial and spatio-temporal classifiers. The evaluation was conducted using five folds cross-validation. The classifiers were compared using confusion matrix, overall classification rate and Dice similarity coefficient.

The experiments demonstrated the following. First, temporal information alone is insufficient in obtaining good classification performance, as all the temporal algorithms tested performed poorly. Second, spatial information is crucial for achieving good classification performance, as the CNN (best spatial classifier) performed significantly better than the LSTM-FCN (best temporal classifier). Third, temporal information can contribute to the overall classification perfor-

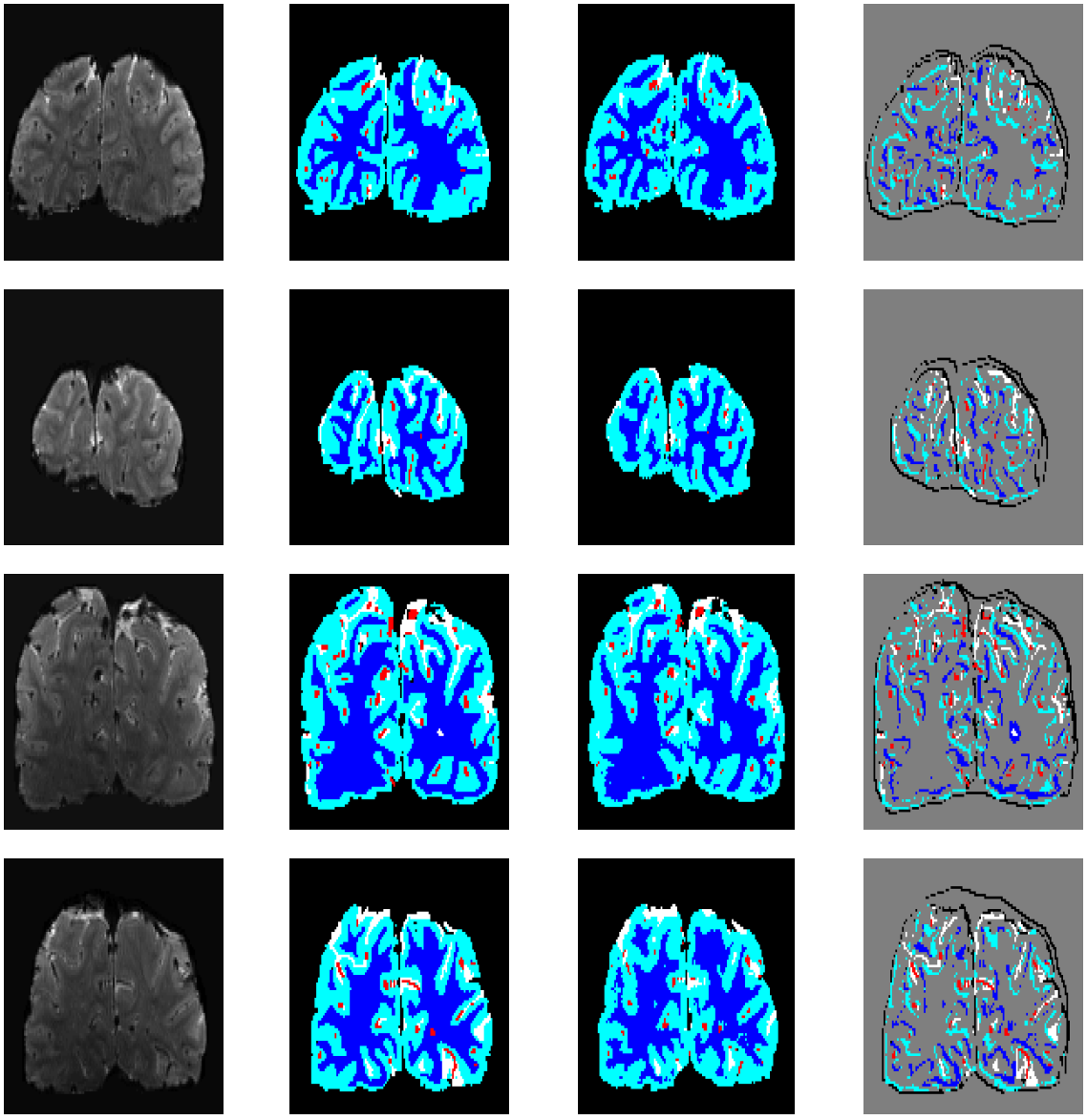


Figure 5.2: Segmentation results for GM (cyan), WM (blue), BV (red), CSF (white), NB (black) in the test set of fold 3. Column 1: Mean image of the fMRI input. Column 2: Ground-truth. Column 3: Segmentation result of the proposed method. Column 4: True class of the incorrectly segmented voxels (gray color denotes the correctly predicted voxel).

mance, as the proposed method (spatio-temporal classifier) outperformed all the classifiers tested in this study.

Chapter 6

Conclusion

Chapter contents

6.1 Thesis summary

The main objective of this research is to develop an automated approach for fMRI brain tissue segmentation. To this end, first, we reviewed the existing works on MRI brain tissue segmentation and fMRI signal classification. The shortcomings of applying the existing approaches on fMRI brain tissue segmentation were also analyzed.

Second, we created an fMRI dataset for the purpose of this study. The raw fMRI data were contributed by Puckett *et al.* [8], we then annotated the ground-truth using the ITK-SNAP software. The preprocessing steps performed on the data were motion correction, slice scan time correction, and intensity normalization.

Third, we proposed a novel patch-wise segmentation method based on deep learning for automatic segmentation of brain tissues in fMRI. The proposed method comprises three stages: spatial feature extraction with convolutional neu-

ral network, temporal feature extraction with long short-term memory and brain tissue class prediction with *softmax* classifier. The network was trained using the Adam optimizer with the categorical cross-entropy loss function.

Fourth, we conducted experiments to determine the optimum hyperparameters for the proposed method. The proposed method was also compared with several temporal domain and spatial domain classifiers. The experiments were conducted using five-fold cross-validation.

6.2 Future works

The future directions of this research can be stated as follows. The first direction is to conduct the experiments with different types of fMRI data, for example data acquired using a higher magnetic field strength MRI machine, such as 7 Tesla (7T). fMRI data acquired using a 7T MRI machine will have a lower gray-white contrast. This is due to the smaller difference in T2 relaxation time between gray and white matter at a higher magnetic field strength. However, the 7T MRI machine allows faster imaging, which can increase the number of time points in an fMRI run.

The second direction is to expand the current fMRI dataset with more subjects. By gathering more data from different subjects, five-fold cross-validation can be performed at a subject-level, allowing a more complete evaluation of the methods. Moreover, this strategy can increase the variations of data, which helps the generalization of machine learning algorithms.

The third direction is to explore 3D spatial information in fMRI data. Exploiting the 3D spatial information can help in identifying brain tissues such as blood vessels that are small in the 2D plane (size of 1 to 5 voxels), but have continuity across the third dimension (slice). The fourth direction is to explore semantic approach for fMRI brain tissue segmentation. Semantic approach is becoming more popular as it can utilize the contextual information of the entire image, and has a shorter computation time.

6.3 Concluding remarks

This thesis proposed a novel deep learning method for automatic segmentation of gray matter, white matter, blood vessel, non-brain and cerebrospinal fluid in fMRI images. The proposed method uses a deep long-term recurrent convolutional network, which can utilize the spatio-temporal information that is present in fMRI data. The proposed method achieves a competitive result, achieving an overall classification rate of 84.04% and an average Dice similarity coefficient of 76.99%.

The experiments conducted demonstrate the following. First, deep learning methods outperform classical machine learning algorithms in segmenting brain tissues in fMRI data. Second, temporal information alone is insufficient in achieving good classification performance. However, temporal information is able to boost the overall classification performance, despite the low temporal resolution. Third, segmenting brain tissues in fMRI images are possible without relying on a T_{1w} image. Finally, the progress made in this research indicates that it is possible for computers to reach the segmentation accuracy of human experts in the near future.

References

[1] S. Abdullah, "T1, T2 and PD weighted imaging," 2017. [Online]. Available: www.radiologycafe.com/radiology-trainees/frcr-physics-notes/t1-t2-and-pd-weighted-imaging

[2] W. Zhang, R. Li, H. Deng, L. Wang, W. Lin, S. Ji, and D. Shen, "Deep convolutional neural networks for multi-modality isointense infant brain image segmentation," *NeuroImage*, vol. 108, pp. 214–224, 2015.

[3] P. Moeskops, M. A. Viergever, A. M. Mendrik, L. S. D. Vries, M. J. N. L. Benders, and I. Išgum, "Automatic segmentation of MR brain images with a convolutional neural network," *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1252–1262, 2016.

[4] A. de Brebisson and G. Montana, "Deep neural networks for anatomical brain segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 20–28.

[5] D. Nie, L. Wang, Y. Gao, and D. Sken, "Fully convolutional networks for multi-modality isointense infant brain image segmentation," in *IEEE International Symposium on Biomedical Imaging*, 2016, pp. 1342–1345.

[6] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer Assisted Intervention*, 2015, pp. 234–241.

[7] A. Riaz, M. Asad, S. M. M. R. A. Arif, E. Alonso, D. Dima, P. Corr, and G. Slabaugh, "Deep fMRI: An end-to-end deep network for classification of fMRI data," in *International Symposium on Biomedical Imaging*, 2018, pp. 1419–1422.

[8] A. M. Puckett, K. M. Aquino, P. A. Robinson, M. Breakspear, and M. M. Schira, "The spatiotemporal hemodynamic response function for depth-dependent

functional imaging of human cortex," *NeuroImage*, vol. 139, pp. 240–248, 2016.

[9] F. Karim, S. Majumdar, H. Darabi, and S. Chen, "LSTM fully convolutional networks for time series classification," *IEEE Access*, vol. 6, pp. 1662–1669, 2017.

[10] M. A. Lindquist, "The statistical analysis of fMRI data," *Statistical Science*, vol. 23, no. 4, pp. 439–464, 2008.

[11] A. Gholipour, N. Kehtarnavaz, R. W. Briggs, K. S. Gopinath, W. Ringe, A. Whittemore, S. Cheshkov, and K. Bakhadirov, "Validation of non-rigid registration between functional and anatomical magnetic resonance brain images," *IEEE Transactions on Biomedical Engineering*, vol. 55, no. 2, pp. 563–571, 2008.

[12] C. Studholme, R. T. Constable, and J. S. Duncan, "Accurate alignment of functional EPI data to anatomical MRI using a physics-based distortion model," *IEEE Transactions on Medical Imaging*, vol. 19, no. 11, pp. 1115–1127, 2000.

[13] B. Fischl, "FreeSurfer," *NeuroImage*, vol. 62, no. 2, pp. 774–781, 2012.

[14] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[15] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 6645–6649.

[16] F. J. Ordonez and D. Roggen, "Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 115, pp. 1–25, 2016.

[17] R. Brown, E. Haacke, Y. Cheng, M. Thompson, and R. Venkatesan, *Magnetic resonance imaging: physical principles and sequence design*. John Wiley & Sons, 2014.

[18] S. Huettel, A. Song, and G. McCarthy, *Functional magnetic resonance imaging*. Sinauer Associates, 2004.

[19] P. M. Matthews and P. Jezzard, "Functional magnetic resonance imaging," *Journal of Neurology, Neurosurgery and Psychiatry*, vol. 75, pp. 6–12, 2004.

[20] Z. Akkus, A. Galimzianova, A. Hoogi, and D. L. Rubin, "Deep learning for brain MRI segmentation: State of the art and future directions," *Journal of Digital Imaging*, vol. 30, no. 4, pp. 449–459, 2017.

[21] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. van der Laak, B. van Ginneken, and C. I. Sanchez, "A survey on deep learning in medical image analysis," *Medical Image Analysis*, vol. 42, pp. 60–88, 2017.

[22] S. Bao and A. C. Chung, "Multi-scale structured CNN with label consistency for brain MR image segmentation," *Computer Methods in Biomechanics and Biomedical Engineering: Imaging and Visualization*, vol. 6, no. 1, pp. 113–117, 2016.

[23] Y. Deng, Y. Sun, Y. Zhu, M. Zhu, and K. Yuan, "A strategy of MR brain tissue images ' suggestive annotation based on modified U-net," *ArXiv e-prints*, pp. 1–13, 2018.

[24] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[25] M. Ringner, "What is principal component analysis?" *Nature Biotechnology*, vol. 26, no. 3, pp. 303–304, 2008.

[26] S.-y. Xie, R. Guo, N. F. Li, G. Wang, and H. T. Zhao, "Brain fMRI processing and classification based on combination of PCA and SVM," in *International Joint Conference on Neural Networks*, 2009, pp. 3384–3389.

[27] R. Meszlényi, L. Peska, V. Gál, Z. Vidnyánszky, and K. Buza, "Classification of fMRI data using dynamic time warping based functional connectivity analysis," in *European Signal Processing Conference*, 2016, pp. 245–249.

[28] D. Kuang and L. He, "Classification on ADHD with deep learning," in *International Conference on Cloud Computing and Big Data*, 2014, pp. 27–32.

[29] S. Sarraf and G. Tofighi, "Deep learning-based pipeline to recognize alzheimer disease using fMRI data," in *Future Technologies Conference*, 2016, pp. 816–820.

[30] J. Shlens, "A tutorial on principal component analysis," *ArXiv e-prints*, pp. 1–12, 2014.

[31] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[32] P. Senin, "Dynamic time warping algorithm review," University of Hawaii, Tech. Rep., 2008.

[33] X. Xi, E. Keogh, C. Shelton, L. Wei, and C. A. Ratanamahatana, "Fast time series classification using numerosity reduction," in *International Conference on Machine Learning*, 2006, pp. 1–8.

[34] A. Mueen and E. Keogh, "Extracting optimal performance from dynamic time warping," in *International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 2129–2130.

[35] K. C. Hutten, "Brain tissue segmentation using sub-millimetre functional magnetic resonance imaging," honours thesis, University of Wollongong, 2014.

[36] K. F. Tan, "Automatic classification of human brain tissues with functional magnetic resonance imaging," honours thesis, University of Wollongong, 2016.

[37] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. John Wiley and Sons, 2012.

[38] P. A. Yushkevich, J. Piven, H. C. Hazlett, R. G. Smith, S. Ho, J. C. Gee, and G. Gerig, "User-guided 3D active contour segmentation of anatomical structures: Significantly improved efficiency and reliability," *NeuroImage*, vol. 31, no. 3, pp. 1116–1128, 2006.

[39] T. R. Oakes, T. Johnstone, K. S. O. Walsh, L. L. Greischar, A. L. Alexander, A. S. Fox, and R. J. Davidson, "Comparison of fMRI motion correction software tools," *NeuroImage*, vol. 28, no. 3, pp. 529–543, 2005.

[40] R. Sladky, K. J. Friston, J. Trostl, R. Cunnington, E. Moser, and C. Windischberger, "Slice-timing effects and their correction in functional MRI," *NeuroImage*, vol. 58, no. 2, pp. 588–594, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.neuroimage.2011.06.078

[41] I. B. Mohamad and D. Usman, "Standardization and its effects on k-means clustering algorithm," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 6, no. 17, pp. 3299–3303, 2013.

[42] J. Donahue, L. Anne Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 677–691, 2016.

[43] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278 – 2324, 1998.

[44] R. Pascanu, D. Tour, T. Mikolov, and D. Tour, "On the difficulty of training recurrent neural networks," in *International Conference on Machine Learning*, 2013, pp. 1310–1318.

[45] A. Graves, "Generating sequences with recurrent neural networks," *ArXiv e-prints*, 2013.

[46] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015, pp. 1–13.

[47] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[48] F. Chollet, "Keras," 2015. [Online]. Available: https://keras.io

[49] J. Bergstra, D. Yamins, and D. D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," *International Conference on Machine Learning*, pp. 115–123, 2013.